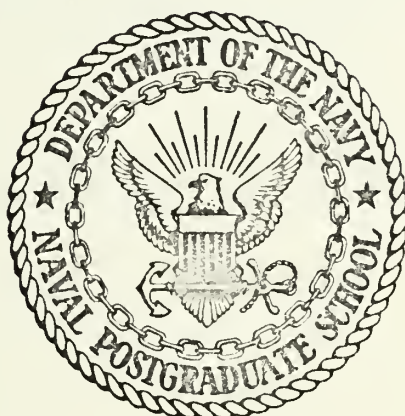


FINITE ELEMENT SOLUTION FOR
AXISYMMETRIC TRANSIENT THERMAL STRESSES

Manouchehr Bakhshandehpour

NAVAL POSTGRADUATE SCHOOL

Monterey, California



THESIS

FINITE ELEMENT SOLUTION FOR
AXISYMMETRIC TRANSIENT THERMAL STRESSES

by

Manouchehr Bakhshandehpour

Thesis Advisor:

R. E. Newton

June 1972

Approved for public release; distribution unlimited.

LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIF. 93940

Finite Element Solution for
Axisymmetric Transient Thermal Stresses

by

Manouchehr Bakhshandehpour
Lieutenant, Imperial Iranian Navy
B.S., Italian Naval Academy, 1960

Submitted in partial fulfillment of the
requirements for the degrees of

MECHANICAL ENGINEER
and
MASTER OF SCIENCE IN MECHANICAL ENGINEERING

from the
NAVAL POSTGRADUATE SCHOOL
June 1972

ABSTRACT

A finite element formulation for solving axisymmetric transient heat conduction and thermal stress problems is developed in this thesis. The governing equations of uncoupled, linear, isotropic thermoelasticity are discretized using quadratic isoparametric elements. A FORTRAN IV program, using double precision arithmetic, is presented. Compact storage techniques for banded symmetric matrices are used.

Comparisons between exact and computer solutions demonstrate close agreement for a number of test problems. Detailed instructions for using the program are included.

TABLE OF CONTENTS

I.	INTRODUCTION -----	12
II.	FINITE ELEMENT FORMULATION OF HEAT CONDUCTION IN AXISYMMETRIC BODIES -----	14
	A. METHOD OF FORMULATION -----	14
	B. THERMAL BOUNDARY CONDITIONS -----	16
	C. EXACT TIME SOLUTION WITH SPATIAL DISCRETIZATION -----	18
	D. TIME INTEGRATION -----	20
	E. ESTIMATION OF EXTREME EIGENVALUES -----	26
III.	ONE-DIMENSIONAL HEAT CONDUCTION -----	29
IV.	STRESS PROBLEM -----	32
	A. STIFFNESS MATRIX -----	33
	B. THERMAL LOAD VECTOR -----	34
	C. PRESSURE LOAD VECTOR -----	36
	D. CENTRIFUGAL LOAD VECTOR -----	37
	E. STRUCTURAL BOUNDARY CONDITIONS -----	38
	F. SYSTEM EQUATION SOLVER -----	39
	G. PRINCIPLE OF SUPERPOSITION -----	39
	H. STRESS EVALUATION -----	39
V.	ONE-DIMENSIONAL TRANSIENT STRESSES -----	41
VI.	TEST PROBLEMS -----	42
VII.	CONCLUSIONS AND RECOMMENDATIONS -----	54
	APPENDIX A: APPLICABLE FORMULAS AND EQUATIONS -----	56
	APPENDIX B: LIST OF THE PROGRAM -----	59
	APPENDIX C: USER'S MANUAL -----	135

APPENDIX D: PROGRAMMING -----	160
LIST OF REFERENCES -----	171
INITIAL DISTRIBUTION LIST -----	172
DD FORM 1473 -----	173

LIST OF FIGURES

1.	Slab with One Face Insulated and Another in Contact with Fluid -----	29
2.	Quadrilateral Element Representation -----	33
3.	Boundary Element Under Pressure -----	36
4.	Two Radial Elements Representation in Thick Cylinder -----	43
5.	Five Radial Elements Representation in Thick Cylinder -----	43
6.	Stresses in Thick Cylinder Under Thermal Loading -----	44
7.	Stresses in Thick Cylinder Under Uniform Internal Pressure -----	45
8.	Stresses in Rotating Thick Cylinder -----	47
9.	Hollow Semi-sphere 32 Elements Representation -----	48
10.	Thermal Stresses in Hollow Sphere -----	49
11.	Nozzle Geometry -----	50
12.	Fluid Temperature-Time Histories -----	52
13.	Element Representation in Nozzle -----	53
14.	Longitudinal Cross Section -----	137
15.	Subdivision into Elements -----	138
16.	Element and Node Numbering -----	139
17.	Entry Temperature-Time Variation -----	151
18.	Deck Set-up for Using CHECK Program -----	155
19.	Deck Set-up for Using AXITTS Program -----	157
20.	Deck Set-up for Obtaining a Listing of the Program AXITTS -----	157
21.	Deck Set-up for Obtaining a Punched Deck of the Program AXITTS -----	158

22. Functional Flow Chart of CANDY ----- 162

23. Fluid Node Representation for Inside Flow ----- 163

24. Functional Flow Chart of STIFF ----- 167

25. Functional Flow Chart of FØRMF ----- 168

LIST OF TABLES

I.	Attenuation Factor Comparison, Fourth Order Runge-Kutta Algorithm -----	23
II.	Attenuation Factor Comparison, Trapezoidal Integration -----	25
III.	Effect of Using Irons' Correction at Every 10 Steps of Integration -----	27
IV.	Comparison of One-dimensional Transient Temperatures -----	31
V.	Units for Input Data -----	136
VI.	Maximum Values for Program Parameters -----	170

LIST OF SYMBOLS

Note: A single underline is used to denote a column vector and a double underline denotes a rectangular matrix. The symbols used in computer program are described in the beginning of Appendix B.

a	Entrance fluid cross-sectional area
$\underline{\underline{A}}$	Linear combination of $\underline{\underline{C}}$ and $\underline{\underline{Y}}$ matrices
\underline{b}	A constant vector
$\underline{\underline{B}}$	Standard rectangular strain-displacement matrix
$\underline{\underline{C}}$	Thermal capacitance matrix
c	Specific heat
$\underline{\underline{D}}$	Standard elasticity matrix
E	Young's modulus of elasticity
e	Superscript designating element contribution
\underline{F}	Load vector
\underline{F}_T^e	Element thermal load vector
\underline{F}_P^e	Element pressure load vector
\underline{F}_C^e	Element centrifugal load vector
$\underline{\underline{G}}$	Linear combination of $\underline{\underline{C}}$ and $\underline{\underline{Y}}$ matrices
h	Surface heat transfer coefficient
$\underline{\underline{I}}$	Identity matrix
$\underline{\underline{J}}$	Jacobian coordinate transformation matrix
$\underline{\underline{K}}$	System stiffness matrix
$\underline{\underline{K}}^e$	Element stiffness matrix
k	Thermal conductivity
L	Thickness of slab

ℓ	Arc length along the side of quadrilateral
N_i	Shape function
m	Total number of nodes
n	Outward normal or number of nodes per element
P	Pressure
R	Radial coordinate
S	Surface area
\underline{T}	Nodal temperature vector
T	Temperature or, when used as a superscript, transpose of a matrix
T_{avg}	Average temperature
T_f	Fluid temperature (used in one-dimensional example)
\underline{U}	A vector defined as $\langle 1 \ 1 \ 1 \ 0 \rangle^T$
u	Radial displacement
\underline{v}	Right-hand side vector in conduction equation
V	Volume
W	Work done by loads
$\underline{\underline{W}}$	Modal matrix
w	Axial displacement
\underline{w}	Eigenvector
$\underline{\underline{Y}}, \underline{\underline{Y}}^+$	Thermal admittance matrix
y	Dependent variable
y_{ij}^*	Element ij of the matrix $\underline{\underline{Y}}^*$
$[\]$	Matrix representation
$\langle \ \rangle$	Row vector
$\bar{\nabla}$	Gradient operator
α	Coefficient of thermal expansion
$\underline{\beta}$	Constant coefficient vector

$\underline{\delta}$	Nodal displacement vector
$\underline{\delta}^e$	Element displacement vector
$\underline{\epsilon}$	Strain vector
μ	Eigenvalue of one-dimensional transient temperature solution
$\underline{\epsilon}_0$	Thermal strain vector
$\underline{\epsilon}^e$	Element strain vector
η	Local element coordinate
ν	Poisson's ratio
λ	Eigenvalue
ρ	Material density
$\underline{\sigma}$	Stress vector
ξ	Local element coordinate
τ	Time
$\Delta\tau$	Step size of numerical time integration
τ_{RZ}	Shearing stress component
θ	Fluid temperature, or angle
$\underline{\underline{\Lambda}}$	Spectral matrix
Ω	Speed of rotation

ACKNOWLEDGEMENTS

The author would like to express his gratitude to the Imperial Iranian Navy for having made it possible for him to undertake the course of graduate study leading to the present text.

To Dr. Robert E. Newton, Professor of Mechanical Engineering, the author wishes to express deep appreciation for his inspiring advice and guidance. Without his diligence and patience this study could not have been accomplished in its present form. The author is obliged to Dr. Gilles Cantin, Professor of Mechanical Engineering, for his generous advice during the course of study and computer programming. Thanks are also due to the personnel at the computer center of Naval Postgraduate School.

Finally the author should thank his wife for her patience and understanding throughout his work at this institution.

I. INTRODUCTION

Thermal stresses have become increasingly important in engineering practice during recent years. In power generation higher cycle temperatures and use of nuclear fission are largely responsible for this trend. This thesis describes a computer program for finding temperatures and stresses in bodies having axisymmetric geometry and loading. The governing equations are those of isotropic, uncoupled, quasi-static, linear thermoelasticity. They are discretized by using the finite element method. A FORTRAN IV computer program using double-precision arithmetic has been written to solve problems of the following kinds.

A. TEMPERATURE PROBLEMS

The transient temperature vector, evaluated at the nodal points, may be obtained for an axisymmetric body with the combination of insulated, convection, or constant temperature boundary conditions. For the convection thermal boundary condition, however, we may have fluid flowing with entry temperature prescribed as a linear function of time (RAMP). The program can handle up to 15 different ramps, each having a different flow velocity, and with discontinuities between successive ramps.

B. STRESS PROBLEMS

The program will generate load vectors for pressure loading, centrifugal loading, and axial force. Provision is

made for direct input of one additional load vector. Stresses may be found for any combination of these loadings.

C. THERMAL STRESS PROBLEMS

Thermal stresses may be found for as many as 20 different temperature vectors which may be output of the temperature problem or direct input. In short, in this part any combination of the temperature and stress problems may be used.

II. FINITE ELEMENT FORMULATION OF HEAT CONDUCTION IN AXISYMMETRIC BODIES

A. METHOD OF FORMULATION

For bodies of revolution under axisymmetric loading the mathematical problems presented are two-dimensional. The governing equation for non-steady heat conduction is

$$\bar{\nabla} \cdot k \bar{\nabla} T = \rho c \dot{T}, \quad (1)$$

where k is the thermal conductivity, ρ the density, c the specific heat, T the temperature and $\bar{\nabla}$ the gradient operator. The superior dot denotes a time derivative.

Applying Galerkin's principle [1] gives

$$\int_V N_i \bar{\nabla} \cdot k \bar{\nabla} T \, dV = \int_V \rho c N_i \dot{T} \, dV, \quad (2)$$

where the integral is over the volume V of the conducting body and N_i is a "shape function" used in representing the temperature distribution. If S is the surface of the body and n the outward normal to surface, then using Gauss' theorem we can write

$$\int_V \bar{\nabla} \cdot (N_i k \bar{\nabla} T) \, dV = \int_S N_i k \frac{\partial T}{\partial n} \, dS. \quad (3)$$

Since

$$\begin{aligned} \int_V \bar{\nabla} \cdot (N_i k \bar{\nabla} T) \, dV &= \int_V (\bar{\nabla} N_i) \cdot (k \bar{\nabla} T) \, dV \\ &+ \int_V N_i \bar{\nabla} \cdot (k \bar{\nabla} T) \, dV, \end{aligned} \quad (4)$$

we can combine Eqs. 2, 3 and 4 to get

$$\int_V \rho c N_i \dot{T} dV + \int_V (\bar{\nabla} N_i) \cdot (k \bar{\nabla} T) dV = \int_S N_i k \frac{\partial T}{\partial n} dS. \quad (5)$$

Each node of the solid region has a separate discretized linear equation calculated from Eq. 5 using the appropriate shape function N_i . Thus each of the volume integrals on the left hand side of Eq. 5 yields a square coefficient matrix in the assembled set of equations. Calculation of these matrices is a standard process. Details are given by Zienkiewicz [1].

The discretized set of equations takes the form

$$\underline{\underline{C}} \underline{\dot{T}} + \underline{\underline{Y}} \underline{T} = \underline{v}, \quad (6)^\vee$$

where there is a term by term correspondence with Eq. 5. The real symmetric matrices $\underline{\underline{C}}$ and $\underline{\underline{Y}}$ represent, respectively, the thermal capacitance and thermal admittance. The elements of the vector \underline{T} are nodal temperatures. The vector \underline{v} , discussed in the following section, depends upon the thermal boundary conditions.

In the present development piecewise constant material properties have been assumed, i.e., k, ρ and c are constant within each element, but may vary from element to element. Also two-dimensional isoparametric elements are used. Applicable equations are summarized in Appendix A.

[✓]In this text a double underline denotes a rectangular matrix and single underline denotes a column vector.

B. THERMAL BOUNDARY CONDITIONS

Thermal boundary conditions affect only those scalar equations of Eq. 6 which correspond to boundary nodes. Accordingly, the vector \underline{v} is sparse. Also, in a single problem it is common to have different thermal boundary conditions on individual portions of the boundary. In what follows the subvectors of \underline{v} (distinguished by individual superscripts) which correspond to separate boundary conditions are treated individually.

1. Insulated

It is clear that for insulated boundary conditions the subvector $\underline{v}^{(1)}$ of the right hand side of Eq. 6 corresponding to this boundary condition is zero, since $\frac{\partial T}{\partial n} = 0$.

2. Convection

The heat transfer mechanism occurs in the interface of the solid and fluid. If the fluid temperature is θ and the heat transfer coefficient is h , then equating heat conducted away from the surface to the efflux from the solid [2] gives

$$-k \left(\frac{\partial T}{\partial n} \right)_S = h(T - \theta), \quad (7)$$

where the subscript S means that the derivative is evaluated at the surface.

For constant h :

$$\int_S N_i k \frac{\partial T}{\partial n} dS = h \int_S N_i (\theta - T) dS. \quad (8)$$

In what follows the fluid temperature θ is taken to be a specified function of position and time. For purposes of discretization, the fluid temperature is specified at a discrete number of fluid "nodes." If θ_j represents the fluid temperature at fluid node j , then the fluid temperature along the boundary may be represented by

$$\theta = \sum N_j \theta_j, \quad (9)$$

where the N_j are one-dimensional forms of the shape functions used for the solid. Substitution in Eq. 8 gives

$$\int_S N_i k \frac{\partial T}{\partial n} dS = \sum_j y_{ij}^* (\theta_j - T_j), \quad (10)$$

where the summation extends over the surface nodes and the coefficients y_{ij}^* are given by

$$y_{ij}^* = h \int_S N_i N_j dS. \quad (11)$$

Assembling the contributions from Eq. 10, the subvector $v^{(2)}$ for the convection boundary condition may be written

$$v^{(2)} = \underline{Y}^* \underline{\theta}. \quad (12)$$

The contributions $-\sum y_{ij}^* T_j$ from Eq. 10 are included by augmenting the matrix \underline{Y} (see Eq. 13 below).

3. Constant Temperature

If θ represents the constant temperature desired at the wetted surface, then we can use the convection boundary condition and replace h by a big number (say 10^{20}). Since h is very large, then for thermal equilibrium the temperature

T at the surface will be forced to equal θ . So the subvector $\underline{v}^{(3)}$ for the portion corresponding to the constant temperature boundary condition can be obtained from Eq. 12.

Upon the application of these boundary conditions in a single problem, the right-hand side vector \underline{v} will be combined from the corresponding subvectors and the finite element discretized equation becomes

$$\underline{C} \dot{\underline{T}} + \underline{Y}^+ \underline{T} = \underline{v}, \quad (13)$$

where $\underline{Y}^+ = \underline{Y} + \underline{Y}^*$.

C. EXACT TIME SOLUTION WITH SPATIAL DISCRETIZATION

We consider only the solution of Eq. 13 for $\underline{v} = \text{constant}$ with $\underline{T} = \underline{a}$ at time $\tau = 0$. Let \underline{T}_s be a particular solution (steady state) with $\dot{\underline{T}}_s = 0$ so that

$$\underline{T}_s = (\underline{Y}^+)^{-1} \underline{v}. \quad (14)$$

For the homogeneous equation

$$\underline{C} \dot{\underline{T}} + \underline{Y}^+ \underline{T} = 0 \quad (15)$$

the assumption $\underline{T} = \underline{w} \exp(-\lambda\tau)$, where \underline{w} is a vector and λ is a scalar, yields the form

$$\underline{Y}^+ \underline{w} = \lambda \underline{C} \underline{w}. \quad (16)$$

It is apparent that Eq. 16 defines an eigenvalue problem.

Let $\underline{\Lambda}$ be the spectral matrix and \underline{W} the modal matrix with normalization according to

$$\underline{W}^T \underline{C} \underline{W} = \underline{I}, \quad (17)$$

where \underline{I} is the identity matrix of the same order as \underline{C} . Now let

$$\underline{T} = \underline{W} \exp(-\underline{\Lambda}\tau) \underline{b} \quad (18)$$

where \underline{b} is a constant vector. Substituting this in Eq. 15 gives

$$\underline{Y}^+ \underline{W} \exp(-\underline{\Lambda}\tau) \underline{b} = \underline{C} \underline{W} \underline{\Lambda} \exp(-\underline{\Lambda}\tau) \underline{b}. \quad (19)$$

Now Eq. 19 is satisfied for all \underline{b} if

$$\underline{W}^T \underline{Y}^+ \underline{W} = \underline{\Lambda}, \quad (19')$$

and this is guaranteed to be satisfied since $\underline{\Lambda}$ and \underline{W} are spectral and modal matrices for the eigenvalue problem of Eq. 16 with \underline{W} normalized according to Eq. 17.

Returning to the original problem (Eq. 13), the complete solution may be written as

$$\underline{T} = \underline{W} (\underline{\beta} + \exp(-\underline{\Lambda}\tau) \underline{b}) \quad (20)$$

where

$$\underline{T}_s = \underline{W} \underline{\beta}, \quad (21)$$

and $\underline{\beta}$ is a constant vector. Now $\underline{\beta}$ may be found (using Eq. 14) to be

$$\underline{\beta} = \underline{\Lambda}^{-1} \underline{W}^T \underline{v}. \quad (22)$$

Substituting this result into Eq. 20 and using the initial condition gives the result

$$\underline{b} = \underline{W}^{-1} \underline{a} - \underline{\Lambda}^{-1} \underline{W}^T \underline{v}. \quad (23)$$

The general solution of Eq. 13 may thus be written as

$$\underline{T} = \underline{W}(\underline{I} - \exp(-\underline{\Lambda}\tau)) \underline{\Lambda}^{-1} \underline{W}^T \underline{v} + \underline{W} \exp(-\underline{\Lambda}\tau) \underline{W}^{-1} \underline{a}. \quad (24)$$

For purposes of the present program non-zero components of vector \underline{v} are to be specified as piecewise linear functions of time. During each segment of time history of \underline{v} an analytical solution of Eq. 13 is possible in the form of a particular solution plus a complementary solution such as Eq. 20. At each node the corresponding time variation of temperature will consist of a linear part contributed by the particular solution and a sum of n terms representing the complementary part. Each of these n terms decays exponentially with a separate time constant. In principle it is a straightforward process to find each particular solution and accompanying complementary solution.

Contemplated problems may typically have from 10 to 40 segments required to represent the piecewise linear variation of \underline{v} . The number of body nodes n will be of the order of 100 or more. In view of the number of particular solutions required, each accompanied by an individual complementary solution of the form given by Eq. 20, it was concluded that a numerical solution of Eq. 13 would be considerably more economical than an analytic one such as that given by Eq. 24.

D. TIME INTEGRATION

In this section the relative merits of the Runge-Kutta and trapezoidal methods of time integration are discussed. Since either of these methods will give an exact result if the solution is a linear function of time, investigation

is confined to performance on a single scalar equation

$$\dot{y} + \lambda y = 0 \quad (25)$$

whose solution $y = y_0 \exp(-\lambda\tau)$ is of the same form as the components of the complementary solution (Eq. 20).

1. Runge-Kutta Method

A method introduced by Runge and subsequently elaborated by Heun and Kutta [3] is widely used for the numerical solution of first order ordinary differential equations. This algorithm prescribes a sequence of calculations for determining the ordinate y_{i+1} at time $\tau_{i+1} = \tau_i + \Delta\tau$ in terms of y_i and values of \dot{y} at intermediate and end points of the interval $\Delta\tau$. The fourth-order form, which requires four evaluations of \dot{y} , gives for Eq. 25 the result

$$\frac{y_{i+1}}{y_i} = 1 - \lambda\Delta\tau + \frac{(\lambda\Delta\tau)^2}{2!} - \frac{(\lambda\Delta\tau)^3}{3!} + \frac{(\lambda\Delta\tau)^4}{4!} . \quad (26)$$

The right-hand side of Eq. 26 represents the first five terms of the Taylor expansion of the exact solution

$(y_{i+1}/y_i = \exp(-\lambda\Delta\tau))$ so we may conclude that the relative error in each time step is less than modulus of the next term: $(\lambda\Delta\tau)^5/5!$.

In addition to providing the apparent prospect for high precision indicated by this error bound, the Runge-Kutta method also permits changes of time increment during the integration process without requiring additional computationally expensive matrix decompositions. The attractiveness of these two features dictated a thorough exploration of the potential of this method for the present application. The

is confined to performance on a single scalar equation

$$\dot{y} + \lambda y = 0 \quad (25)$$

whose solution $y = y_0 \exp(-\lambda\tau)$ is of the same form as the components of the complementary solution (Eq. 20).

1. Runge-Kutta Method

A method introduced by Runge and subsequently elaborated by Heun and Kutta [3] is widely used for the numerical solution of first order ordinary differential equations. This algorithm prescribes a sequence of calculations for determining the ordinate y_{i+1} at time $\tau_{i+1} = \tau_i + \Delta\tau$ in terms of y_i and values of \dot{y} at intermediate and end points of the interval $\Delta\tau$. The fourth-order form, which requires four evaluations of \dot{y} , gives for Eq. 25 the result

$$\frac{y_{i+1}}{y_i} = 1 - \lambda\Delta\tau + \frac{(\lambda\Delta\tau)^2}{2!} - \frac{(\lambda\Delta\tau)^3}{3!} + \frac{(\lambda\Delta\tau)^4}{4!} . \quad (26)$$

The right-hand side of Eq. 26 represents the first five terms of the Taylor expansion of the exact solution

($y_{i+1}/y_i = \exp(-\lambda\Delta\tau)$) so we may conclude that the relative error in each time step is less than modulus of the next term: $(\lambda\Delta\tau)^5/5!$.

In addition to providing the apparent prospect for high precision indicated by this error bound, the Runge-Kutta method also permits changes of time increment during the integration process without requiring additional computationally expensive matrix decompositions. The attractiveness of these two features dictated a thorough exploration of the potential of this method for the present application. The

disqualifying defect which emerged after studying a number of examples is readily appreciated from examination of Table I. For values of $\lambda\Delta\tau$ less than 0.5 it is apparent that Runge-Kutta scheme affords acceptable engineering accuracy. However, when the method is applied to solution of Eq. 13 we must deal with a number of λ 's equal to the number of nodes (see Eq. 20). This number may be greater than 100 and the ratio of the largest λ to the smallest may easily exceed 1000. Although the solution is dominated by the contributions of the eigenvectors corresponding to the smaller λ s, it is clear that the solution will be unstable if the largest $\lambda\Delta\tau$ exceeds about 2.7. Because an unacceptably small $\Delta\tau$ is required in typical problems, the Runge-Kutta method was rejected.

2. Trapezoidal Method

The trapezoidal method estimates y_{i+1} from the formula

$$y_{i+1} = y_i + \frac{\Delta\tau}{2} (\dot{y}_i + \dot{y}_{i+1}). \quad (27)$$

Substituting for \dot{y}_i and \dot{y}_{i+1} from Eq. 25 and rearranging gives

$$\frac{y_{i+1}}{y_i} = \frac{2 - \lambda\Delta\tau}{2 + \lambda\Delta\tau}. \quad (28)$$

Series expansion of the right-hand side provides an error bound (per step):

$$(\lambda\Delta\tau)^3/12.$$

From the point of view of the size of $\lambda\Delta\tau$ this method has no stability limit, but has slow attenuation with alteration

TABLE I
ATTENUATION FACTOR COMPARISON
Fourth Order Runge-Kutta Algorithm

$\lambda\Delta\tau$	y_{i+1}/y_i (Runge-Kutta)	$\exp(-\lambda\Delta\tau)$ (Exact)
.0001	.9999	.9999
.001	.9990	.9990
.01	.9901	.9901
.1	.9048	.9048
.2	.8187	.8187
.5	.6068	.6065
1.0	.3750	.3679
2.0	.3333	.1353
2.5	.6484	.0821
3.0	1.3750	.0498
4.0	5.0000	.0183
8.0	110.3333	.0003
10.0	291.0000	.0000
20.0	5514.3333	.0000
50.0	240784.3333	.0000
100.0	4004901.0000	.0000

in sign for large $\lambda\Delta\tau$. Table II shows this behavior.

Since a wide usable range of $\lambda\Delta\tau$ is essential and the stability of trapezoidal integration is guaranteed, this method is chosen for the present program.

Applying the trapezoidal algorithm to Eq. 13 yields

$$\underline{\underline{A}} \underline{T}^{i+1} = \underline{\underline{G}} \underline{T}^i + \frac{\Delta\tau}{2} (\underline{v}^{i+1} + \underline{v}^i), \quad (29)$$

where

$$\underline{\underline{A}} = \underline{\underline{C}} + \frac{\Delta\tau}{2} \underline{Y}^+,$$

$$\underline{\underline{G}} = \underline{\underline{C}} - \frac{\Delta\tau}{2} \underline{Y}^+.$$

and the superscripts denote evaluation at discrete time intervals $\Delta\tau$. If m is the order of the capacitance and admittance matrices, $\underline{\underline{C}}$ and \underline{Y} , then once a certain step size $\Delta\tau$ is chosen, it requires $m^3/3$ operations to perform the needed triangular decomposition of $\underline{\underline{A}}$. Thus, for large m , a change of step size $\Delta\tau$ becomes costly from the point of view of computer time. Accordingly, in the present program only one time step size is used throughout each problem.

Also, for assuring sufficiently rapid attenuation of the components corresponding to the large $\lambda\Delta\tau$, the following correction is utilized.

3. Irons' Correction

Irons proposed a scheme [4] for augmenting the attenuation of the contributions of those eigenvectors for which $\lambda\Delta\tau$ is large. Define

TABLE II
ATTENUATION FACTOR COMPARISON
Trapezoidal Integration

$\lambda\Delta\tau$	y_{i+1}/y_i (Trapezoidal)	$e^{-\lambda\Delta\tau}$ (Exact)
.0001	.9999	.9999
.001	.9990	.9990
.01	.9901	.9901
.1	.9048	.9048
.2	.8182	.8187
.3	.7391	.7408
.5	.6000	.6065
1.0	.3333	.3679
2.0	.0000	.1353
2.5	-.1111	.0821
3.0	-.2000	.0498
4.0	-.3333	.0183
8.0	-.6000	.0003
10.0	-.6667	.0000
20.0	-.8182	.0000
50.0	-.9231	.0000
100.0	-.9608	.0000

$$y_i^* = .25 y_{i-1} + .5 y_i + .25 y_{i+1}, \quad (30)$$

where y_i and y_{i+1} are obtained from y_{i-1} by trapezoidal integration.

In the program presented in Appendix B, Eq. 30 is used after every 10 steps of time integration. Table III shows the resulting modifications.

E. ESTIMATION OF EXTREME EIGENVALUES

Analytic results for one-dimensional heat conduction give, for an eigenvalue,

$$\lambda = \frac{\pi^2 k}{4\rho c} \frac{1}{d^2}, \quad (31)$$

where d is the distance between points of extreme temperature and zero temperature.

If we use this to estimate the smallest λ in cylindrical coordinates, two modifications are recommended.

1. Assume that the point of zero temperature is in the fluid at a distance from the wall equal to k/h , where h is the surface heat transfer coefficient.
2. If there are two approximately orthogonal paths for heat flow from the (single) maximum temperature point, then replace $1/d^2$ in the above formula by

$$\frac{1}{d^2} = \frac{1}{d_{\min}^2} + \frac{1}{d_{\max}^2}. \quad (32)$$

For estimating the largest λ , the surface heat transfer coefficient has no significant effect. We may continue to

TABLE III
EFFECTS OF USING IRONS' CORRECTION
AFTER 10 STEPS OF INTEGRATION

$\lambda\Delta\tau$	$\exp(-10\lambda\Delta\tau)$ Exact	y_{10}/y_0 Trapezoidal	y_{10}^*/y_0 Corrected
0.00010	0.99900	0.99900	0.99900
0.00020	0.99800	0.99800	0.99800
0.00100	0.99005	0.99005	0.99005
0.01000	0.90484	0.90484	0.90486
0.10000	0.36788	0.36757	0.36849
0.20000	0.13534	0.13443	0.13579
0.30000	0.04979	0.04866	0.04978
0.50000	0.00674	0.00605	0.00645
1.00000	0.00005	0.00002	0.00002
2.00000	0.00000	0.00000	0.00000
4.00000	0.00000	0.00002	-0.00001
8.00000	0.00000	0.00605	-0.00040
10.00000	0.00000	0.01734	-0.00072
20.00000	0.00000	0.13443	-0.00136
50.00000	0.00000	0.44914	-0.00072
100.00000	0.00000	0.67028	-0.00027

use the same formula for $\frac{1}{d^2}$, but now consider only the smallest element and take

$$d_{\min} = \frac{\text{length of smallest side}}{4},$$
$$d_{\max} = \frac{\text{length of largest side}}{4}.$$
(32')

A comparison of estimates based on Eq. 31, 32, 32' with the exact solution of the eigenvalue problem has been carried out for several examples. Based on these comparisons, it is believed that these estimates are sufficiently accurate for choosing a time step and estimating the time of occurrence of the extreme stresses.

III. ONE-DIMENSIONAL HEAT CONDUCTION

For comparison of numerical (time) integration methods, studies of one-dimensional heat conduction were made. In this section numerical results for trapezoidal time integration using Irons' correction are compared with the exact transient temperature solution.

Consider a flat slab of thickness L with conductivity k , density ρ , specific heat c , zero initial temperature, one face insulated and the other in contact with fluid at temperature T_f (Fig. 1). The surface heat transfer is h . The exact transient heat conduction solution is available [5] as

$$T = T_f \left\{ 1 - 2 \sum_{i=1}^{\infty} \frac{\sin \mu_i L \cos \mu_i x}{\mu_i L + \sin \mu_i L \cos \mu_i L} \cdot e^{-\mu_i^2 \frac{k}{\rho c} \tau} \right\} \quad (33)$$

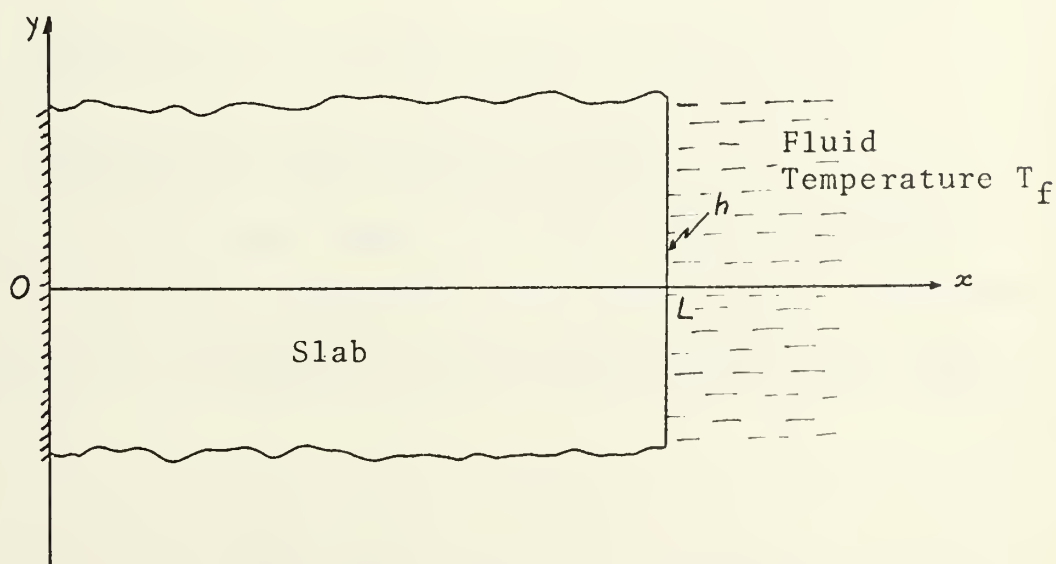


Figure 1. Slab with One Face Insulated and Another in Contact with Fluid.

where $k/\rho c$ is the diffusivity and the μ_i are the solutions of the transcendental equation

$$\tan \mu L = \frac{hL}{k} \frac{1}{\mu L} . \quad (34)$$

For the finite element comparisons we subdivide the distance L into m one-dimensional 3-noded elements and use the corresponding isoparametric shape functions. (The working equations, shape functions and the element capacitance and admittance matrices are given in Appendix A.)

In the course of this investigation separate computer programs were written to evaluate nodal transient temperatures using the following methods:

- (a) exact transient temperature solution, Eq. 33;
- (b) exact time solution with spatial discretization (section II.C, Eq. 24);
- (c) Runge-Kutta time integration (section II.D, part a);
- (d) trapezoidal time integration (section II.D, part b);
- (e) trapezoidal time integration with Irons' correction (section II.D, part c).

For an initial step change of fluid temperature from zero to 1, transient temperatures have been found. For these comparisons the parameters (in consistent units) were taken to be:

$$L = 8, \rho = 25, k = 8, c = 5 \text{ and } h = 5$$

The distance L was subdivided into $m = 3$ elements. For the present purpose, comparison is confined to the exact solution (item (a)), and the finally adopted system (item (e)).

In Table IV, temperatures at the two faces ($x = 0$, $x = 8$) are compared for various times. The trapezoidal integration has been performed using the constant time increment unity and the Irons' correction is applied after every 10 increments.

It is believed that Table IV demonstrates that the numerical integration method gives adequate accuracy for engineering applications.

TABLE IV
COMPARISON OF ONE-DIMENSIONAL TRANSIENT TEMPERATURES

Method	x	Time=0	Time=10	Time=20	Time=30	Time=50	Time=70
Exact	0.	.00000	.00000	.00000	.00002	.00095	.00560
Trapezoidal with Irons'✓	0.	.00000	.00028	.00007	.00066	.00207	.00619
Exact	8.	.00000	.38431	.47684	.53284	.60264	.64685
Trapezoidal with Irons'✓	8.	.00000	.38720	.47716	.53262	.60256	.64686

✓ Irons' correction is used after every 10 steps of trapezoidal integration.

IV. STRESS PROBLEM

For bodies of revolution deformed symmetrically under axisymmetric loading, the stress distribution is two-dimensional. Since deformation is symmetric about the axis of revolution, cylindrical coordinates (R, Z, θ) are used. It follows that the stress components are independent of the angle θ and all derivatives with respect to θ are zero. Also the components of shearing stress $\tau_{R\theta}$ and $\tau_{Z\theta}$ vanish on account of the symmetry. But since any radial displacement induces a strain ϵ_θ in the circumferential direction, this non-zero component of strain and the three in-plane components $(\epsilon_Z, \epsilon_R, \gamma_{RZ})$, complete the state of strain at a point in any axisymmetric situation. Hence the state of stress for an axisymmetric body under axisymmetric loading is given by

$$\underline{\sigma} = \langle \sigma_Z \ \sigma_R \ \sigma_\theta \ \tau_{RZ} \rangle^T . \quad (35)$$

In this chapter the stiffness matrix of an axisymmetric body and the thermal, pressure, and centrifugal load vectors are formulated and, finally, evaluation of stresses at a point is discussed. The treatment closely follows that of Zienkiewicz [1] and this reference should be consulted for further details.

A. STIFFNESS MATRIX

The elements used are bodies of revolution (about the Z axis). For analysis it is sufficient to describe the cross-section in the R,Z plane. In Fig. 2 such an element and the local ξ, η coordinates are shown.

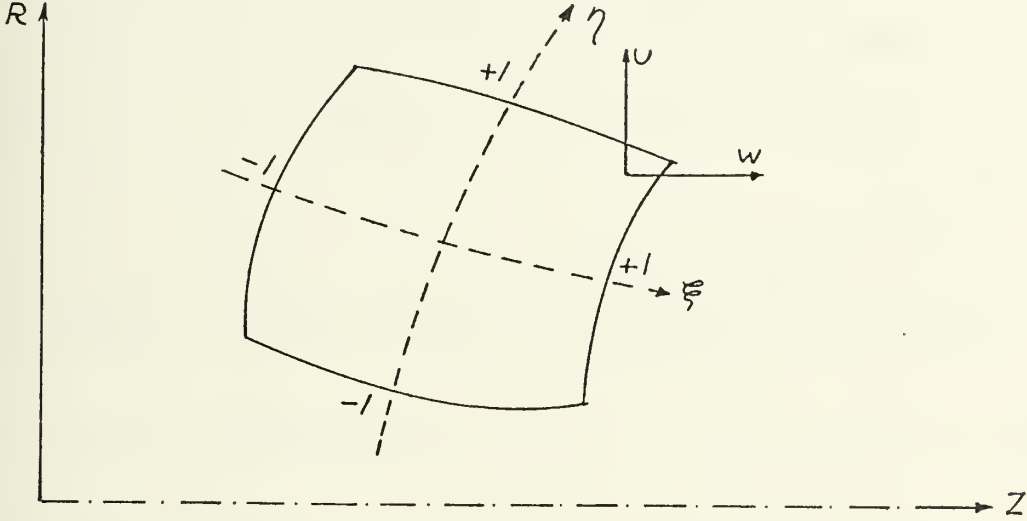


Figure 2. Quadrilateral Element Representation.

If u and w are the displacement components at a point in the directions of R and Z respectively, then these displacement components may be defined in terms of the nodal displacements by the appropriate isoparametric shape functions as

$$u = \sum_{i=1}^8 N_i u_i, \quad w = \sum_{i=1}^8 N_i w_i, \quad (36)$$

where N_i , a function of ξ and η , is the shape function for element node i and u_i, w_i , are the nodal displacement components. The strain-displacement relations [6] can now be used to obtain the components of the strain vector. Thus

$$\underline{\underline{\epsilon}} = \underline{\underline{B}} \underline{\underline{\delta}} , \quad (37)$$

where $\underline{\underline{B}}$ is the standard rectangular strain-displacement matrix of any finite element formulation, a function of the local coordinates ξ and η , and $\underline{\underline{\delta}}$ is the vector of nodal displacement. (See Appendix A, part 3, where the applicable formulas and useful equations are summarized).

If the elasticity matrix for an isotropic material is $\underline{\underline{D}}$, then the stress vector $\underline{\underline{\sigma}}$ at a point is given by

$$\underline{\underline{\sigma}} = \underline{\underline{D}} \underline{\underline{\epsilon}} . \quad (38)$$

Now, by evaluation of the total strain energy in the element, the element stiffness matrix can readily be obtained as

$$\underline{\underline{K}}^e = \int \underline{\underline{B}}^T \underline{\underline{D}} \underline{\underline{B}} dV , \quad (39)$$

where the integration extends over the volume of the element.

In the present program the upper triangle of each element stiffness matrix is evaluated by numerical integration using four Gauss points within the range of ξ and of η [1]. The element contribution is immediately placed in the system stiffness matrix, which is stored in banded form.

B. THERMAL LOAD VECTOR

If we denote T as the difference between local temperature and reference temperature, then the thermal strain $\underline{\underline{\epsilon}}_0$ is given as

$$\underline{\underline{\epsilon}}_0 = \underline{\underline{U}} \alpha T, \quad (40)$$

where

$$\underline{U} = \langle 1 \ 1 \ 1 \ 0 \rangle^T$$

and α is the coefficient of thermal expansion.

The thermal load vector \underline{F}_T^e is given by Zienkiewicz [1] as

$$\underline{F}_T^e = \int \underline{B}^T \underline{D} \underline{\epsilon}_0^e dV . \quad (41)$$

From the point of view of the numerical evaluation it is interesting to note, however, that the product $\underline{D} \underline{\epsilon}_0^e$ in Eq. 41 will reduce to

$$\underline{D} \underline{\epsilon}_0^e = \frac{E \alpha T}{1-2\nu} \underline{U} , \quad (42)$$

where E is the modulus of elasticity and ν is Poisson's ratio. Thus

$$\underline{F}_T^e = \frac{E \alpha}{1-2\nu} \int T \underline{B}^T \underline{U} dV , \quad (43)$$

where $T = \sum_{i=1}^n N_i T_i$ and n is the number of nodes in each element.

In the attached computer program in Appendix B the advantage of the simplicity of Eq. 42 has been utilized. Also, since \underline{B} has some zero components, in the process of multiplication of $\underline{B}^T \underline{U}$, simply the addition of the appropriate non-zero components of each column of the \underline{B}^T has been performed. Finally, Eq. 43 has been integrated with four Gauss points.

C. PRESSURE LOAD VECTOR

Consider a quadrilateral element as in Fig. 3 on the boundary of the axisymmetric cross-section where constant pressure P is applied. The infinitesimal force dF due to the normal pressure acting on the inner infinitesimal circumferential surface dS is

$$dF = PdS = 2 \pi RP \, d\ell , \quad (44)$$

where $d\ell$ is the infinitesimal length along the side of quadrilateral.

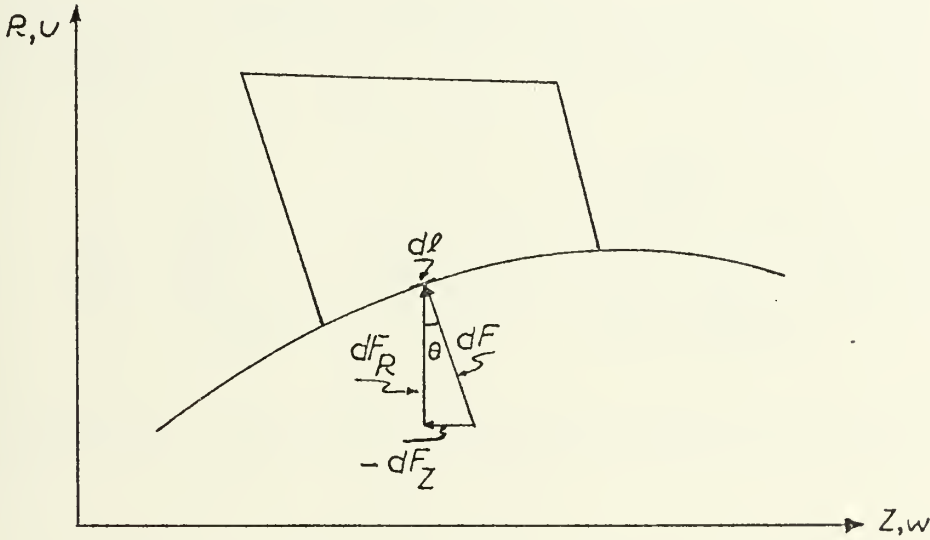


Figure 3. Boundary Element Under Pressure.

Let F_R and F_Z be the components of the pressure force in the R and Z directions respectively, then

$$dF_R = dF \cos \theta, \quad dF_Z = -dF \sin \theta , \quad (45)$$

where $\sin \theta = \frac{dR}{d\ell}$ and $\cos \theta = \frac{dZ}{d\ell}$. Therefore

$$dF_R = (2 \pi RP) dZ , \quad dF_Z = -(2 \pi RP) dR. \quad (46)$$

Since the total work W done by the normal force F is equal to the sum of the work done in R and Z directions,

$$\begin{aligned} W &= \int u \, dF_R + \int w \, dF_Z \\ &= 2\pi P (\int R \, u \, dZ - \int R \, w \, dR). \end{aligned} \quad (47)$$

Now, by using the appropriate shape functions, each component of Eq. 47 may be defined in terms of the nodal values.

Since

$$W = (\underline{\delta}^e)^T \underline{F}_p^e, \quad (48)$$

where \underline{F}_p^e is the element pressure load vector
vector contributed by node j is explicitly given as

$$\underline{F}_{jp}^e = \begin{bmatrix} F_R \\ F_Z \end{bmatrix}_{jp}^e = 2\pi P \int_{-1}^1 (\sum N_i R_i) \begin{bmatrix} \sum \frac{\partial N_i}{\partial \xi} Z_i \\ -\sum \frac{\partial N_i}{\partial \xi} R_i \end{bmatrix} N_j \, d\xi, \quad (49)$$

where N_j in Eq. 49 is the appropriate shape function for node j .

D. CENTRIFUGAL LOAD VECTOR

Refer again to Fig. 2 and assume that the body is rotating about the Z axis with angular speed Ω . Then the centrifugal force per unit volume at a point distant R from the Z axis will be $\rho\Omega^2 R$, where ρ is the density of the material. The work done in this case is

$$W = \int_V \rho\Omega^2 R \, u \, dV. \quad (50)$$

For constant $\rho\Omega^2$ the corresponding element centrifugal load vector is readily obtained by evaluation of the components of the integral in Eq. 50 in terms of the nodal variables, i.e.,

$$F_{j_c}^e = 2\pi\rho\Omega^2 \int_{-1}^1 \int_{-1}^1 (\sum N_i R_i)^2 \det \underline{J} N_j d\xi d\eta, \quad (51)$$

where $\det \underline{J}$ is the determinant of the Jacobian coordinate transformation matrix (see Appendix A).

E. STRUCTURAL BOUNDARY CONDITIONS

The structural boundary conditions implemented in the program are:

- (a) one or more nodes prevented from moving axially;
- (b) one or more nodes prevented from moving radially;
- (c) the right-hand end cross-sectional plane remains plane and the transmitted axial force is specified. Hereinafter this will be referred to as the plane-end boundary condition.

The computer program presented in Appendix B has the capability of handling any combination of the above mentioned structural boundary conditions. For boundary conditions of the types (a) and (b), simply multiplying the corresponding diagonal component of the stiffness matrix by 10^{20} gives zero displacement [8] (for practical purposes). For the boundary condition of type (c) both ends are initially fixed axially for all solutions. An additional solution is obtained for unit axial displacement of one end. The axial force is evaluated for each solution. Superposition is performed by adding the displacement vectors for the given

loadings (thermal plus mechanical), plus an appropriate fraction of the vector found for unit axial displacement. This fraction is chosen so that the resultant axial load has the specified value.

F. SYSTEM EQUATION SOLVER

Once the desired structural boundary conditions are applied, then the problem is to find the nodal displacement vector $\underline{\delta}$, corresponding to a given number of load vectors. We have

$$\underline{K} \underline{\delta} = \underline{F} , \quad (52)$$

where \underline{K} is the system stiffness matrix in banded form and \underline{F} is a load vector. In the present computer program a single Cholesky decomposition is performed on \underline{K} . Then, by a process of forward and back substitution, each load vector is replaced by the corresponding displacement vector.

G. PRINCIPLE OF SUPERPOSITION

Upon the evaluation of the displacement vectors due to the various types of loading, the principle of superposition can be applied on the displacement vectors. On each thermal displacement vector the displacement due to any other type of loading is superimposed and, as the result, the number of displacement vectors is reduced to the number of thermal load vectors.

H. STRESS EVALUATION

From the system displacement vector $\underline{\delta}$, the displacement vector of each element $\underline{\delta}^e$ may be obtained easily. Then the

corresponding element strain vector $\underline{\epsilon}^e$ at any point can be found from

$$\underline{\epsilon}^e = \underline{B} \underline{\delta}^e . \quad (53)$$

Finally, the corresponding element stress vector $\underline{\sigma}^e$ is obtained by

$$\underline{\sigma}^e = \underline{D} (\underline{\epsilon}^e - \underline{\epsilon}_0^e) . \quad (54)$$

Since normally the stresses on the inner and outer surfaces of axisymmetric bodies are desired, in the computer program presented in the Appendix B provision has been made to calculate the stresses at the two Gauss points corresponding to $\xi = \pm \frac{1}{\sqrt{3}}$ on the inner and outer boundaries of each element (where $\eta = \pm 1$).

Upon the evaluation of the stresses at each point the mean stress and the octahedral shearing stress [7] are calculated. The program gives as output the extreme values of these stresses, the R and Z coordinates of the corresponding points, and the times of occurrence.

V. ONE-DIMENSIONAL TRANSIENT STRESSES

In this section a one-dimensional comparison of stresses is made between exact and finite element results. The transient temperature problem is the one previously described in Section III.

If the slab edges are free to translate in the plane of the slab, but are prevented from rotating, the exact solution for thermal stress [9] is

$$\sigma_y = \sigma_z = \frac{E\alpha}{1-\nu} (T_{\text{avg}} - T) , \quad (55)$$

where T is the local temperature (Eq. 33), and

T_{avg} is the average temperature in the slab

If we choose $E = 2$, $\alpha = .50$, and $\nu = 0$ (all in consistent units), then the maximum stress obtained from the exact solution is

$$\sigma_{\text{max}} = -.477786$$

and it occurs at $x = 8$, $\tau = 73$.

Using the finite element technique with trapezoidal time integration and Irons' correction every 10 steps, the maximum stress is found to be

$$\sigma_{\text{max}} = -.477778$$

and it also occurs at $x = 8$, $\tau = 73$, as before.

It is observed that the method chosen gives excellent results.

VI. TEST PROBLEMS

Program integrity and accuracy have been verified by solving a number of test problems. Since stresses, whose evaluation depends upon derivatives of displacements, are known to be less accurate than temperatures, comparisons with exact results are confined to stresses. Individual problems are described below.

A. THICK CYLINDER

Consider a thick cylinder with inside radius 30 inches and outside radius 50 inches and the following material properties.

Modulus of elasticity	$E = 28.9 \times 10^6$	Psi
Poisson's ratio	$\nu = .28$	-
Coefficient of thermal expansion	$\alpha = 7.22 \times 10^{-6}$	1/F°
Thermal conductivity	$k = 28.$	$\frac{\text{Btu}}{\text{hr. ft}^{\circ}\text{F}}$
Density	$\rho = 489.$	Lbm/ft ³
Specific heat	$c = .111$	$\frac{\text{Btu}}{\text{Lbm.}^{\circ}\text{F}}$

An arbitrary length of 25 inches has been selected for the cylinder and it has been subdivided into two different element representations as in Figs. 4 and 5. The plane-end boundary condition with zero axial force is used. The stresses for various types of loading are compared with the corresponding exact analytic solutions as described below.

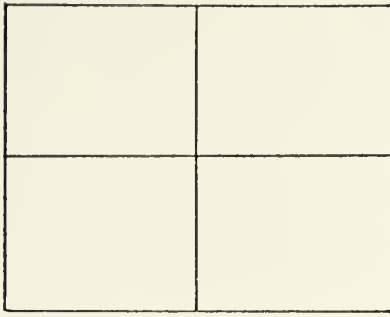


Figure 4. Two Radial Elements Representation of Thick Cylinder.

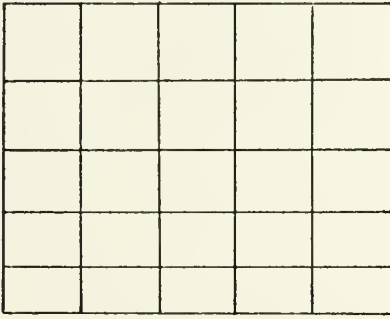


Figure 5. Five Radial Element Representation of Thick Cylinder.

1. Thermal Loading

For a linear variation of the temperature $T = 20R$ the stresses obtained by the finite element method for the above representations are compared with the exact analytic solution in Fig. 6. The τ_{RZ} for this problem clearly is zero and the one obtained by the program was 10^{-9} . The accuracy of the other results is clearly satisfactory.

2. Pressure Loading

A uniform pressure of 1000 psi. acts on the inner surface of the cylinder. Again, τ_{RZ} is zero and the program gives 10^{-10} . In Fig. 7 the other stresses induced by this uniform pressure are compared with the exact solutions. Here also the accuracy of the results, even with two radial elements, is adequate.

KSI.

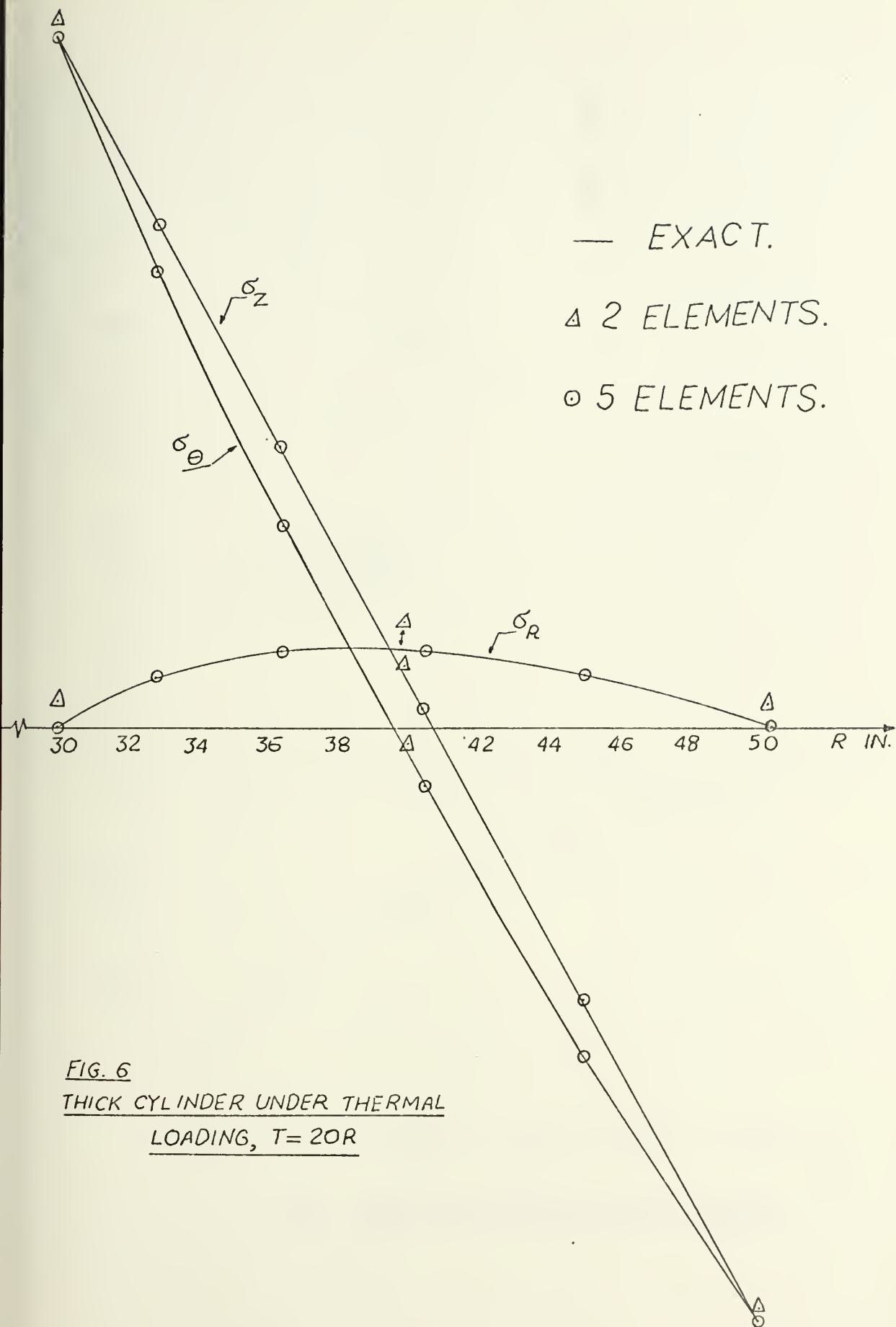


FIG. 6
THICK CYLINDER UNDER THERMAL
LOADING, $T = 20R$

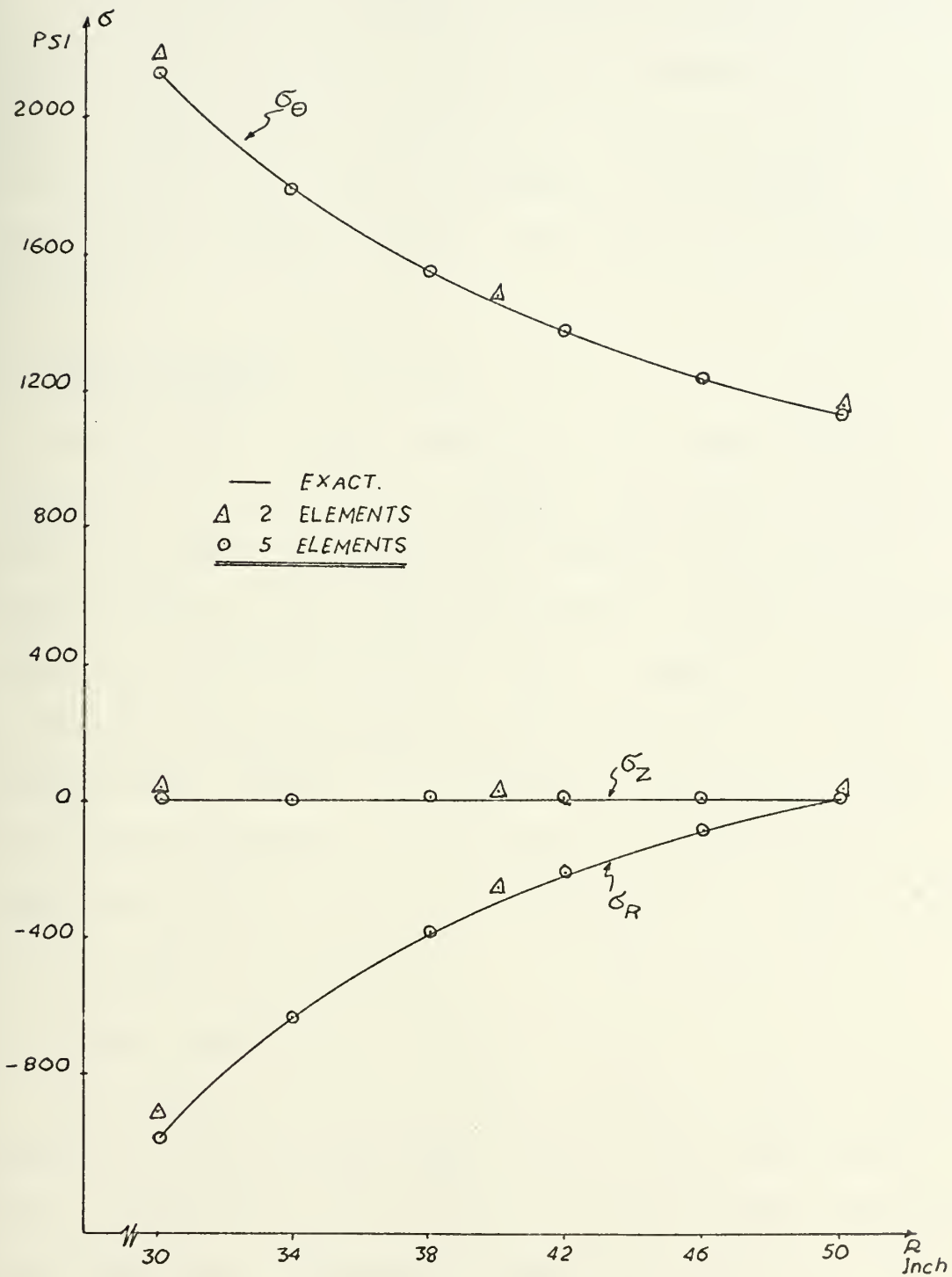


FIG. 7 THICK CYLINDER UNDER INTERNAL PRESSURE.

3. Centrifugal Loading

The speed of rotation has been assumed to be 500 revolutions per minute. The stresses obtained by the program are compared with the analytic solution in Fig. 8. In this case also the results obtained by the program, even with only two radial elements, are very close to the exact solution.

B. HOLLOW SPHERE

We consider a hollow sphere with the same material properties as in the thick cylinder with inside spherical radius 30 inches and outside 50 inches. The loading is thermal with $T = 20 \cdot (\text{spherical radius})$. Symmetry permits using only half of the sphere for the computer analysis. The elements representation is given in Fig. 9. Since the program gives stresses in cylindrical coordinates, these have been transformed to the spherical coordinates for comparison with the exact solution in Fig. 10. The accuracy of the results is noteworthy.

C. THERMAL STRESSES IN NOZZLE

This problem concerns thermal stresses near the intersection of a cylindrical pipe and the spherical vessel. Fig. 11 gives the cross-section of the structure which is to be analysed. The material properties are:

Modulus of elasticity	$E = 29.3 \times 10^6$	Psi
Poisson's ratio	$\nu = .30$	-
Thermal expansion coefficient	$\alpha = 7.6 \times 10^{-6}$	$1/F^\circ$

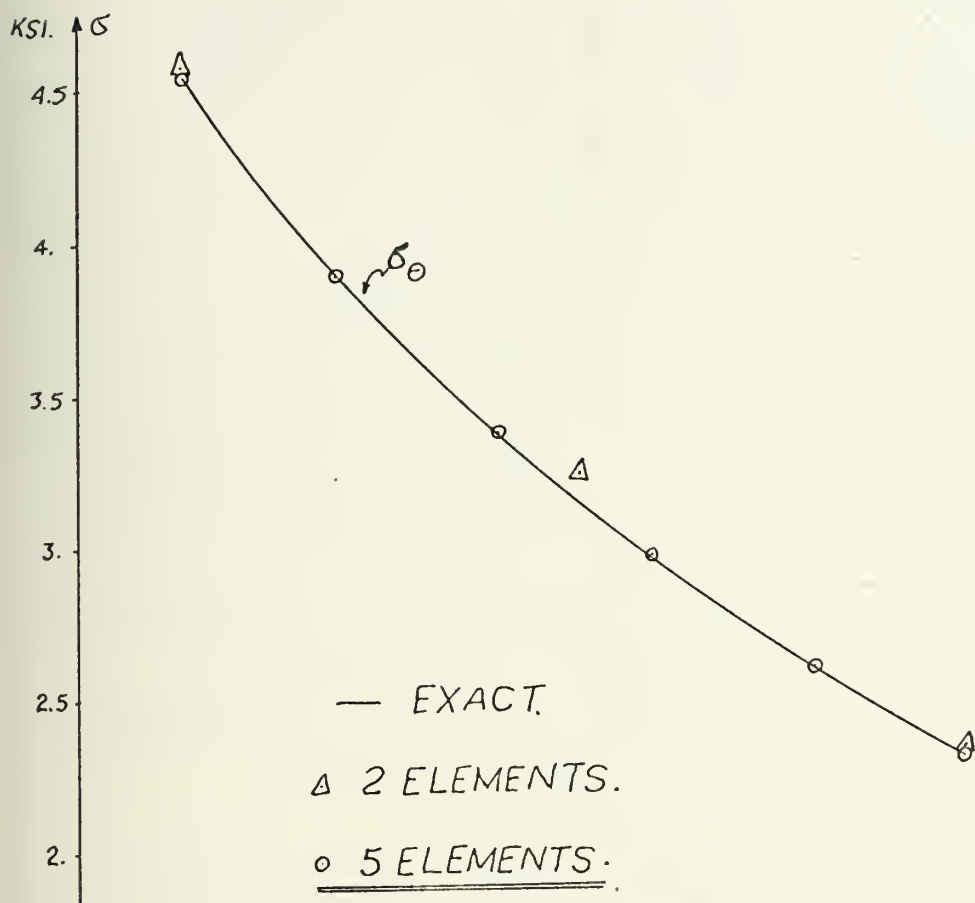
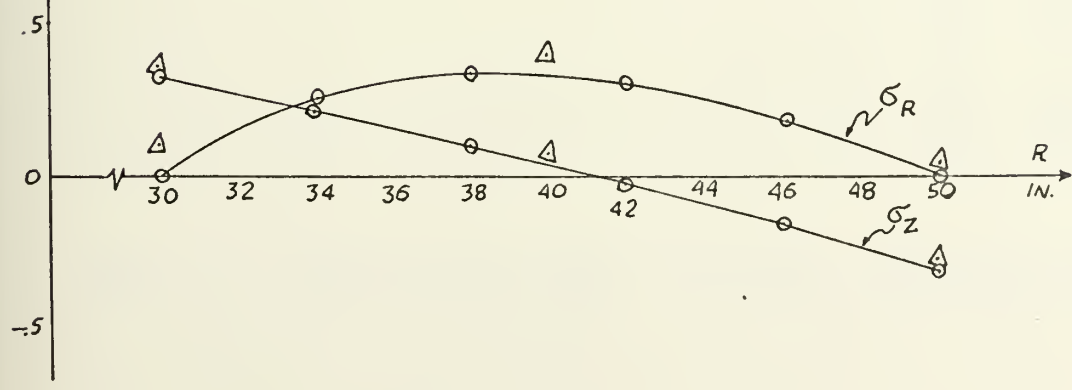


FIG. 8
ROTATING CYLINDER



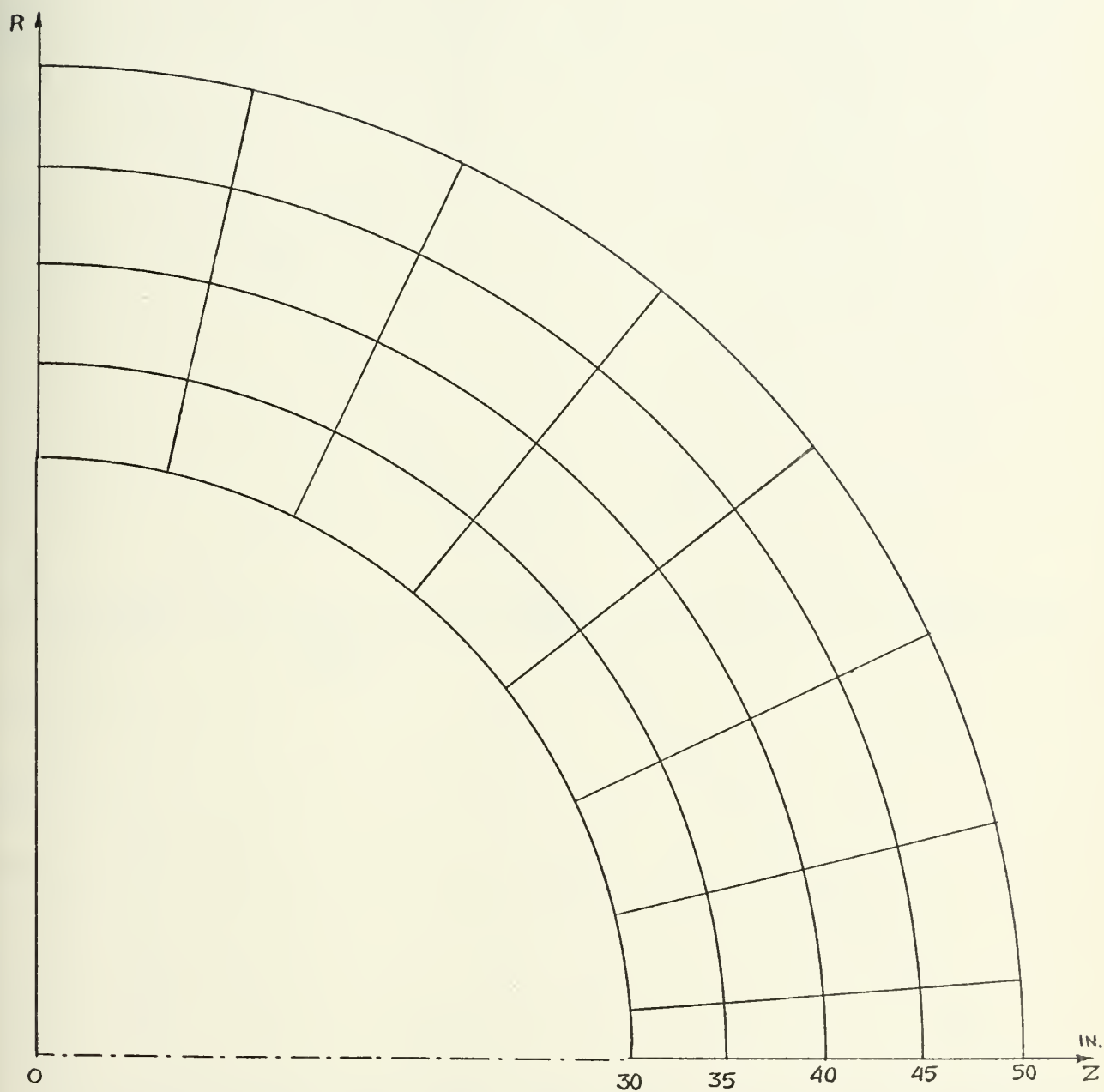


Figure 9. Hollow Semi-sphere 32 Elements Representation.

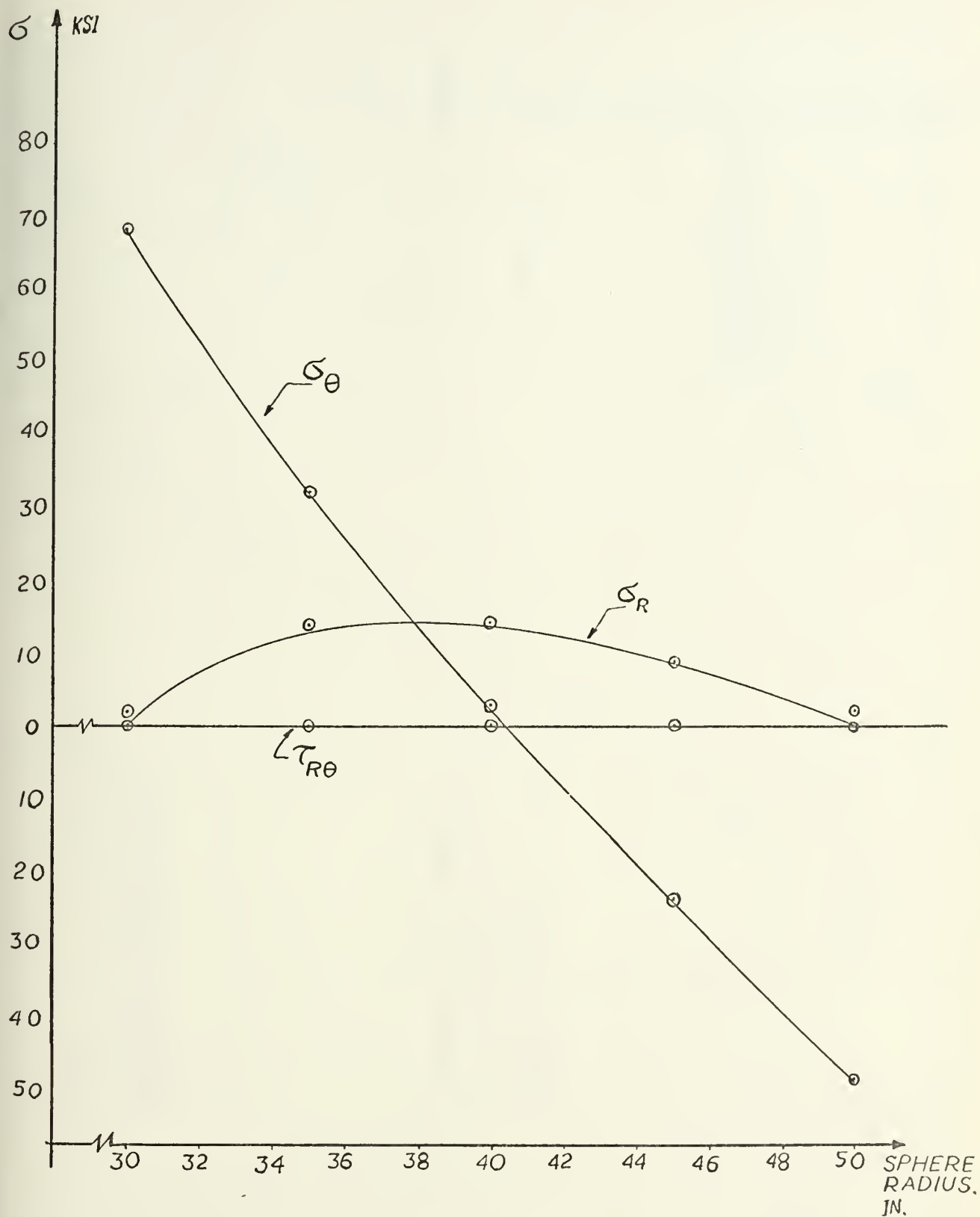
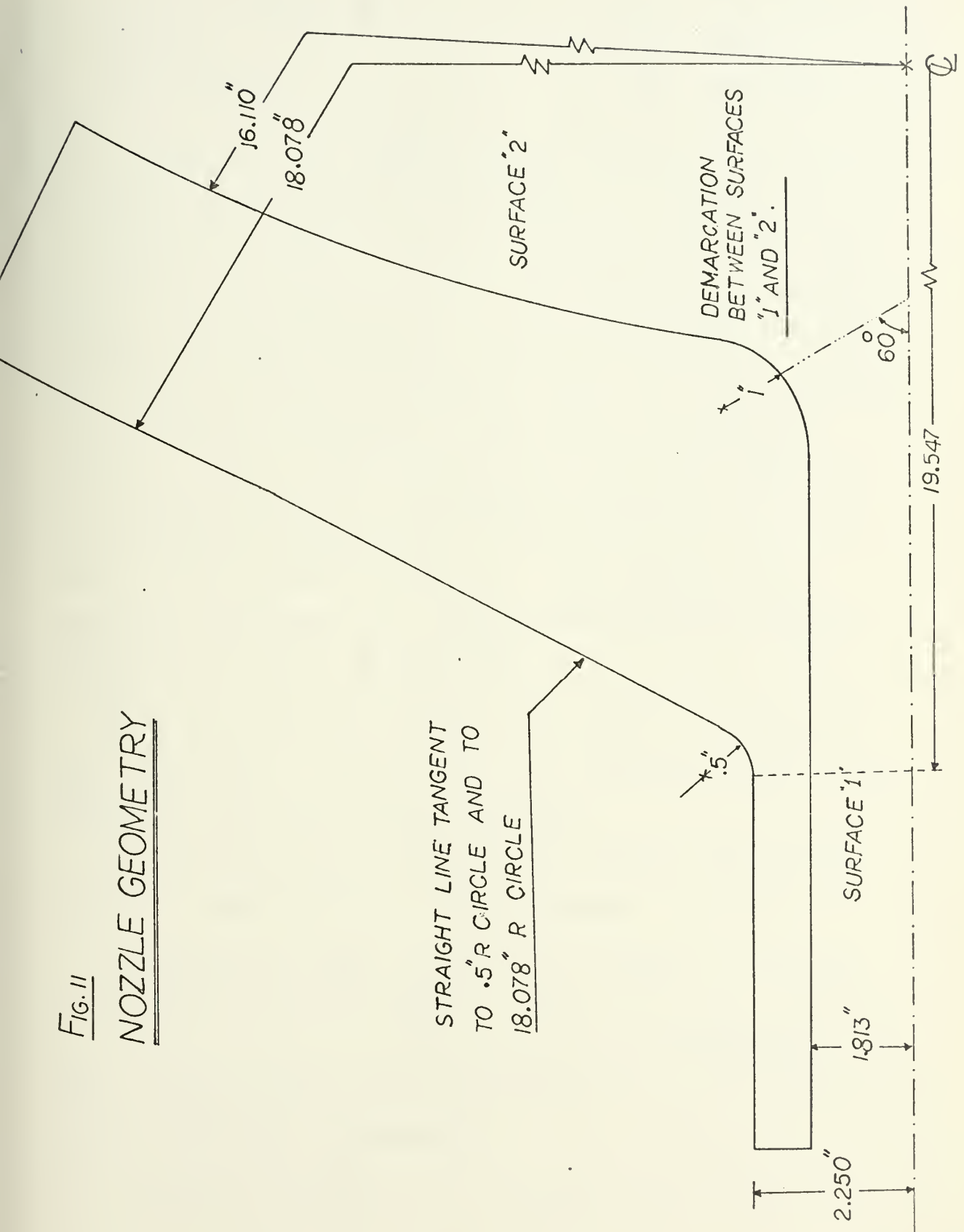


Figure 10. Thermal Stresses in Hollow Sphere
 $T = 20$ (spherical radius).

FIG. 11
NOZZLE GEOMETRY



Thermal conductivity	$k = 10.25$	Btu/ft.hr.°F
Density	$\rho = 530.5$	Lbm/ft ³
Specific heat	$c = .128$	Btu/Lbm°F

The loading results from the thermal transient in the fluid contained in the nozzle. This fluid is in contact with the structure on surface "1" (Fig. 11) and has the entry temperature time variation as in Fig. 12. The fluid in the sphere, which is in contact with the structure on surface "2" (Fig. 11), has the constant temperature 478°F. At $\tau = 0$ structure has a uniform temperature of 478°F and is stress free. Exterior surface of the structure is insulated. The flow velocity past surface 1 is 8.5 ft/sec (inward) and there is no flow past surface 2. The surface heat transfer coefficients are 1393 and 2910 Btu/hr.ft²°F for surfaces "1" and "2" respectively.

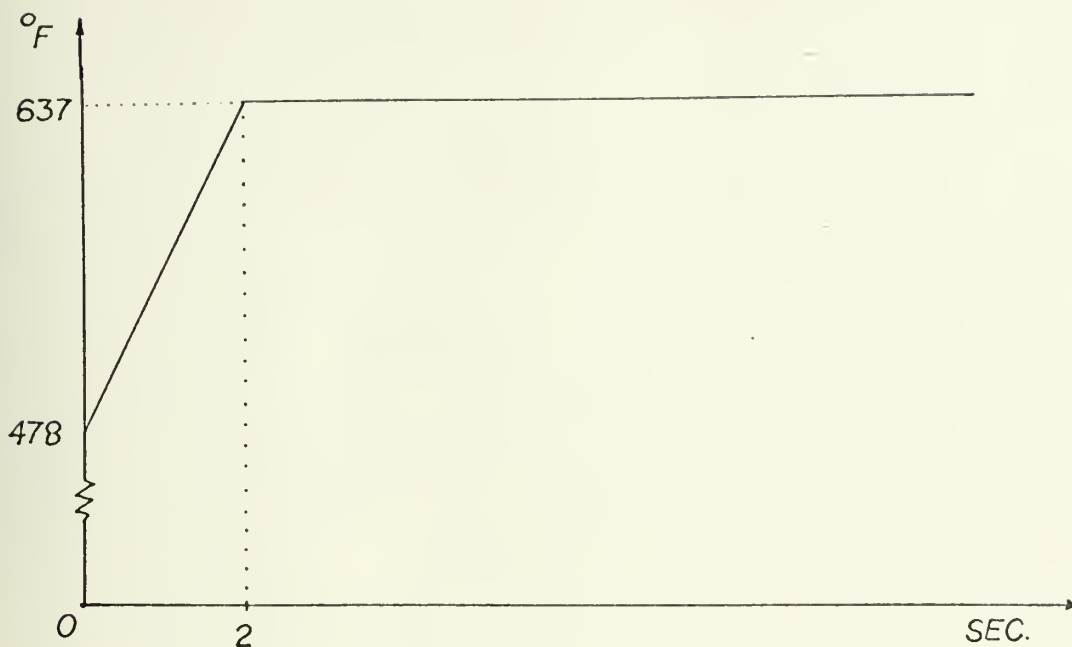
For the structural boundary condition it is assumed that the nodes on the left end cross-section of Fig. 13 are prevented from any axial displacement.

The maximum thermal stresses obtained by the program occur in element 14 of Fig. 13 as follows:

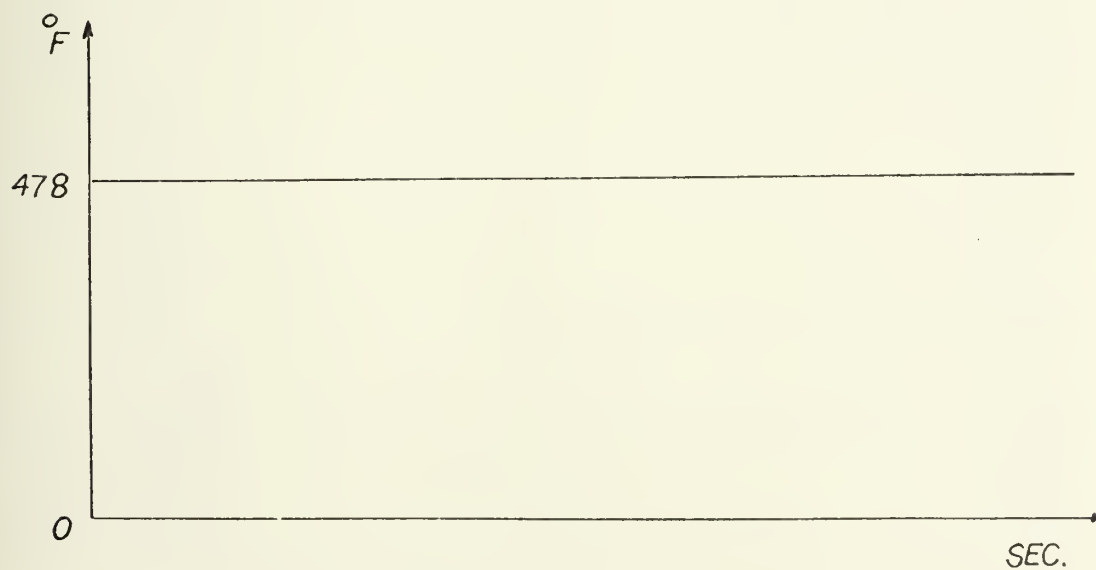
Maximum mean stress = 18.81 ksi. at time = 4 seconds;

Maximum octahedral shearing stress = 11.5 ksi. at time
= 18 seconds.

These results appear to be reasonable, but no suitable comparison solution is available.



VARIATION OF ENTRY FLUID TEMPERATURE
IN NOZZLE. (SURFACE "1".)

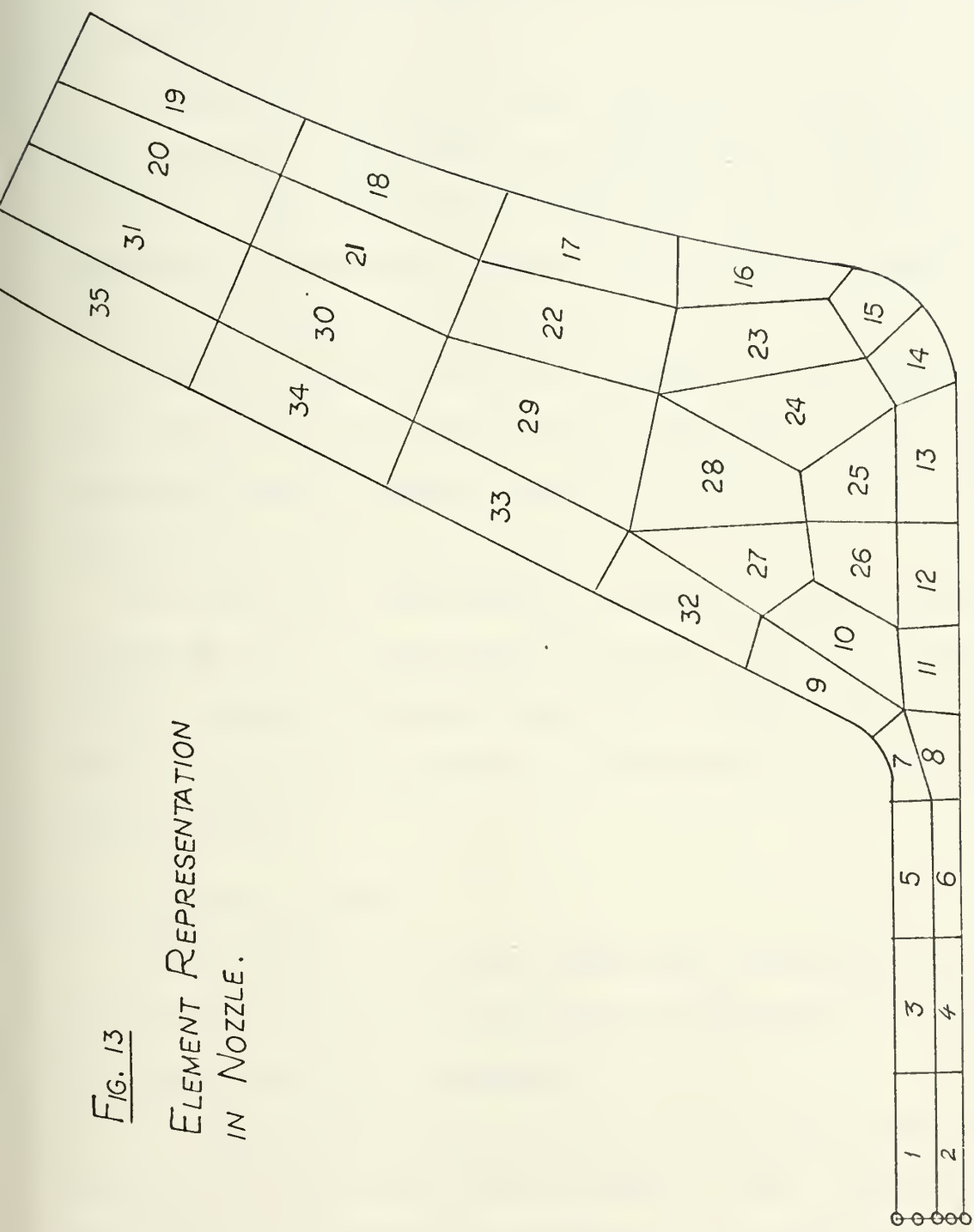


FLUID PAST SURFACE "2".

FIG. 12. Fluid Temperature - Time Histories.

FIG. 13

*ELEMENT REPRESENTATION
IN NOZZLE.*



VII. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

A computer program has been developed for the solution of axisymmetric transient heat conduction and thermal stress problems. The system will accommodate a wide variety of geometric arrangements, thermal and structural boundary conditions, and mechanical loadings.

Although double-precision arithmetic is employed throughout, efficient algorithms for the manipulation and storage of large symmetric banded matrices allow in-core solutions with modest time requirements.

The quadratic isoparametric elements used allow accurate representation of curvilinear boundaries and the stress field. Examples presented show that a small number of elements is generally sufficient to determine stresses with good precision.

B. RECOMMENDATIONS

Incorporation of several additional features would significantly increase the program capabilities. The following extensions are recommended.

1. Material thermal and elastic properties have been assumed constant within each element. Since such properties are generally temperature dependent, provisions should be made for periodically "updating" them during both temperature and stress solutions.

2. The surface heat transfer coefficient, taken as constant in the program, is a function of temperature and flow velocity. Provision should be made to include these effects.

3. The program presently starts every temperature solution with constant initial solid temperature. Provision for an externally specified initial temperature vector should be included.

4. The large quantity of temperature and stress results generated by the program is currently presented as digital printout. Graphical output in the form of two-dimensional contour plots of temperatures and stresses should be provided.

APPENDIX A

APPLICABLE FORMULAS AND EQUATIONS

PART 1

- (a): Two dimensional 8-noded (parabolic) isoparametric shape functions.

Corner nodes:

$$N_i = \frac{1}{4} (1 + \xi_o)(1 + \eta_o)(\xi_o + \eta_o - 1)$$

Mid nodes:

$$\xi_i = 0, \quad N_i = \frac{1}{2} (1 - \xi^2)(1 + \eta_o)$$

$$\eta_i = 0, \quad N_i = \frac{1}{2} (1 + \xi_o)(1 - \eta^2)$$

where

$$\xi_o = \xi \xi_i, \quad \eta_o = \eta \eta_i$$

- (b): Temperature at a point in terms of the nodal temperatures.

$$T = \sum_{i=1}^8 N_i T_i$$

- (c): Coordinates at a point in terms of the nodal coordinates.

$$R = \sum_{i=1}^8 N_i R_i$$

$$Z = \sum_{i=1}^8 N_i Z_i$$

- (d): The Jacobian coordinate transformation matrix.

$$\underline{\underline{J}} = \begin{bmatrix} \frac{\partial Z}{\partial \xi} & \frac{\partial R}{\partial \xi} \\ \frac{\partial Z}{\partial \eta} & \frac{\partial R}{\partial \eta} \end{bmatrix}$$

(e): Element of the finite element capacitance matrix.

$$C_{ij}^e = \rho c \int N_i N_j dV$$

(f): Element of the finite element admittance matrix.

$$Y_{ij}^e = k \int \bar{\nabla} N_i \cdot \bar{\nabla} N_j dV$$

(g): Elemental volume.

$$dV = 2\pi R \det \underline{\underline{J}} d\xi d\eta$$

PART 2

(a): One-dimensional 3-noded (parabolic) isoparametric shape functions.

$$\text{End nodes} \quad N_i = \frac{1}{2} \xi_0 (1 + \xi_0)$$

$$\text{Mid node} \quad N_i = (1 - \xi^2)$$

$$\text{where, again, } \xi_0 = \xi \xi_i$$

(b): One-dimensional capacitance and admittance matrices.

$$\underline{\underline{C}}^e = \frac{\rho c \ell}{30} \begin{bmatrix} 4 & 2 & -1 \\ 2 & 16 & 2 \\ -1 & 2 & 4 \end{bmatrix}, \quad \underline{\underline{Y}}^{+e} = \frac{k}{3\ell} \begin{bmatrix} 7 & -8 & 1 \\ -8 & 16 & -8 \\ 1 & -8 & 7 \end{bmatrix} + \begin{bmatrix} h & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

where ℓ is the length of the element.

(c): The element \underline{v} vector for the one-dimensional case.

$$\underline{v}^e = \langle h T_f \ 0 \ 0 \rangle^T$$

PART 3

(a): Strain-displacement relations.

$$\epsilon_Z = \frac{\partial w}{\partial Z} = \sum \frac{\partial N_i}{\partial Z} w_i$$

$$\epsilon_R = \frac{\partial u}{\partial R} = \sum \frac{\partial N_i}{\partial R} u_i$$

$$\epsilon_\theta = \frac{u}{R} = \frac{\sum N_i U_i}{\sum N_i R_i}$$

$$\dot{\gamma}_{RZ} = \frac{\partial u}{\partial Z} + \frac{\partial w}{\partial R} = \sum \frac{\partial N_i}{\partial Z} U_i + \sum \frac{\partial N_i}{\partial R} w_i$$

where N_i are the same isoparametric shape function as in Part 1, (a).

(b): The B matrix.

$$\underline{\underline{B}} = \begin{bmatrix} 0 & \frac{\partial N_1}{\partial Z} & 0 & \frac{\partial N_2}{\partial Z} & \dots & \frac{\partial N_8}{\partial Z} \\ \frac{\partial N_1}{\partial R} & 0 & \frac{\partial N_2}{\partial Z} & 0 & \dots & 0 \\ \frac{N_1}{R} & 0 & \frac{N_2}{R} & 0 & \dots & 0 \\ \frac{\partial N_1}{\partial Z} & \frac{\partial N_1}{\partial R} & \frac{\partial N_2}{\partial Z} & \frac{\partial N_2}{\partial R} & \dots & \frac{\partial N_8}{\partial R} \end{bmatrix}$$

(c): The element displacement vector.

$$\underline{\underline{\delta}}^e = \langle u_1 \ w_1 \ u_2 \ w_2 \ \dots \ u_8 \ w_8 \rangle^T$$

(d): Elasticity matrix D for an isotropic material.

$$\underline{\underline{D}} = \frac{E(1-\nu)}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1 & \frac{\nu}{1-\nu} & \frac{\nu}{1-\nu} & 0 \\ \frac{\nu}{1-\nu} & 1 & \frac{\nu}{1-\nu} & 0 \\ \frac{\nu}{1-\nu} & \frac{\nu}{1-\nu} & 1 & 0 \\ 0 & 0 & 0 & \frac{1-2\nu}{2(1-\nu)} \end{bmatrix}$$

A. IDENTIFICATION

TITLE

AXITTS
(AXISYMMETRIC TRANSIENT
THERMAL STRESSES.)

COMPLETION DATE
PROGRAMMER

JUNE 1972

ADVISER

LT. M. BAKSHANDEHPOUR
IMPERIAL IRANIAN NAVY
DR. ROBERT E. NEWTON
PROF. MECH. ENG. N.P.G.S.

B. OBJECTIVE

THIS PROGRAM IS USED FOR THE FOLLOWING CASES:

1- TO EVALUATE TRANSIENT TEMPERATURE IN AN AXI-
SYMMETRIC BODY FOR THE COMBINATION OF THE
CONSTANT TEMPERATURE, INSULATED AND CONVEC-
TION THERMAL BOUNDARY CONDITION.

2- TO EVALUATE THE STRESSES IN AN AXISYMMETRIC
BODY FOR ANY COMBINATION OF AXIAL, PRESSURE,
CENTRIFUGAL OR ANY EXTERNALLY SPECIFIED LOAD
VECTOR.

3- TO EVALUATE TRANSIENT THERMAL STRESSES IN AN
AXISYMMETRIC BODY FOR ANY KNOWN, TRANSIENT
TEMPERATURE OR USING THE RESULTS OF STEP 1.
THE COMBINATION OF THERMAL LOADING WITH ANY
OTHER LOADING AS IN STEP 2 MAY BE USED ALSO.

C. METHOD OF CALCULATION

THE FINITE ELEMENT METHOD WITH EIGHT NODDED ISO-
PARAMETRIC ELEMENTS IS APPLIED FOR THE ANALYSIS.

D. USAGE

REFER TO THE (APPENDIX C) OF THE MECHANICAL ENG.
THESIS WRITTEN BY THE PROGRAMMER.

00000030
00000040
00000050
00000060
00000070
00000080
00000090
00000100
00000110
00000120
00000130
00000140
00000150
00000160
00000170
00000180
00000190
00000200
00000210
00000220
00000230
00000240
00000250
00000260
00000270
00000280
00000290
00000300
00000310
00000320
00000330
00000340
00000350
00000360
00000370
00000380
00000390
00000400
00000410
00000420
00000430
00000440
00000450
00000460
00000470
00000480
00000490
00000500

[illegible]

DESCRIPTION OF THE SYMBOLS.

IS THE TOTAL NUMBER OF NODES LONGITUDINALLY FIXED.^a


```

NNRT      TOTAL NUMBER OF NODES RADIALLY FIXED      00000990
NPROB     NUMBER OF PROBLEMS                        00001000
NPASS     NUMBER OF STEPS OF TIME INTEGRATION.      00001010
NPNT      TOTAL NUMBER OF PRESSURE NODES.          00001020
NCFI      ARRAY OF NODES IN CONTACT WITH FLUID     00001030
NCFO      ARRAY OF NODES IN CONTACT WITH FLUID     00001040
OMEGA     NUMBER OF REVOLUTIONS PER MINUTE.        00001050
POI       POISSON'S RATIO.                         00001060
PRES      PRESSURE. THE RADIAL COORDINATES.        00001070
R         ARRAY OF THE SHAPE FUNCTIONS.           00001080
SHP       SPECIFIC HEAT.                         00001090
SHT       SYSTEM OF STIFFNESS MATRIX.             00001100
SSM       ARRAY OF STORED TEMPERATURE VECTORS.     00001110
STOR      MAX. MEAN STRESS. SHEAR STRESS.          00001120
SMAX      MAX. OCTAHEDRAL STRESS.                  00001130
SHMAX     INITIAL TEMPERATURE OF THE BODY.         00001140
TINIT     THERMAL CONDUCTIVITY.                   00001150
TK        TIME WHEN MAX. MEAN STRESS OCCURS.       00001160
TMAXM     TIME WHEN MAX. OCTAHEDRAL SHEAR          00001170
TMAXO     OCCURS. TEMPERATURE.                     00001180
TREF      REFERENCE TEMPERATURE.                   00001190
Z         ARRAY OF THE LONGITUDINAL COORDINATES.  00001200
*****
IMPLICIT  REAL * 8 (A-H,O-Z)
REAL      * 4  ENTERO,ENTERI,ENTIO,ENTII,PLOT
COMMON   /ONE/ AJ(2,2),B(4,16),COPY(37,298),D(4,4),DL(4,4),
1DNZR(2,8),ESM(16,16),F(24,298),SSM(298,66),STOR(20,150)
COMMON   /TWO/ AL(5),DENS(5),DETA(8),DXI(8),E(5),EPSO(4),
1FE(16),GC(4),GX(4),POI(5),R(149),SHP(8),SHT(5),TK(5),Z(149)
COMMON   /TRE/ AXIALF,DTI,EVERY,FORS1,OMEGA,HTCI,HTCO,
1POSI,PRES,Q1,Q2,Q3,SHMAX,SMAX,TIME,TIMEL,TINIT,TMAXM,TREF,
2TMAXO,RMAXM,RMAXO,ZMAXM,ZMAXO,Q4,BIG
COMMON   /FOR/ NN(40,9)
COMMON   /FIV/ NCFI(37),NCFO(37),NNR(37),NRE(37),NNL(37),
1NPN(37)
COMMON   /SIX/ IBAND,IEXT,INTP,IVEC,JVEC,KK,LOAD,NCFIT,NCFOT,
1NCHECK,NET,NDF,NNE,NNLT,NNRT,NPASS,NPROB,NRPPI,NRPPO,
3NNT,NNRE,NPNT,IPLANE,NBAND
EQUIVALENCE (C,SSM) , (Y,SSM(1,18)) , (I,SSM(1,35)) ,

```



```

1(BTE,SSM(1,36)), (TEMP,SSM(1,37)), (V,SSM(1,38)),
2(FT1,SSM(1,39)), (FT2,SSM(1,40)), (CE,SSM(1,41)),
3(YE,SSM(1,42)), (OFF1,SSM(1,43)), (OFF2,SSM(1,44)),
4(DIAG,SSM(1,45)), (BDRY1,SSM(1,46)), (BDRY0,SSM(1,47)),
5(ENTII,SSM(1,48)), (ENTIO,SSM(1,49)), (ENTEI,SSM(1,50)),
6(ENTEO,SSM(1,51)), (VOL,SSM(1,52)), (VLO,SSM(1,53))
C
DIMENSION C(149,33),Y(149,33),T(149),BTE(149),TEMP(149),
1V(149),FT1(90),FT2(90),CE(8,8),YE(8,8),OFF1(37),OFF2(37),
2DIAG(37),BDRY1(16,5),BDRY0(16,5),VOL(36),VLO(36),ENTE0(75),
3ENTEI(75),ENTIO(75),ENTII(75)
C
CALL INPUT
C
TEMPERATURE PROBLEM
C
DO 2 L = 1 , NPROB
C
DO 5 I = 1 , 75
ENTE0(I) = 0.0
ENTEI(I) = 0.0
ENTIO(I) = 0.0
ENTII(I) = 0.0
C
WRITE (6,99) L
99 FORMAT (1H1,/,50X,'PROBLEM NUMBER ',15,/,50X,
120(' ',/),50X,20(' '),//)
CALL PROB
C
IF ((Q4.LT.0.D0).OR.(IEXT.NE.0)) GO TO 3
C
CALL CANDY
DO 1 K = 1 , NPASS
CALL FLOW
CALL FORMV
CALL TEMPER
1 CONTINUE
C
IF (Q2.GT.0.D0) CALL PLOT(ENTIO,ENTE0,0)
IF (Q3.GT.0.D0) CALL PLOT(ENTII,ENTEI,1)
C
IF (Q4.GT.0.D0) GO TO 2

```

```

00001470
00001480
00001490
00001500
00001510
00001520
00001530
00001540
00001550
00001560
00001570
00001580
00001590
00001600
00001610
00001620
00001630
00001640
00001650
00001660
00001670
00001680
00001690
00001700
00001710
00001720
00001730
00001740
00001750
00001760
00001770
00001780
00001790
00001800
00001810
00001820
00001830
00001840
00001850
00001860
00001870
00001880
00001890
00001900
00001910
00001920
00001930
00001940

```



```

1V(149),FT1(90),FT2(90),CF(8,8),YE(8,8),OFF1(37),OFF2(37),
2DIAG(37),BDRY1(16,5),BDRY0(16,5),VOL(36),VLO(36),ENTE0(75),
3ENTE1(75),ENTIO(75),ENTII(75)
4DIMENSION HCO(149), VCO(149)

```

```

C DATA BRIT/'BRIT'/

```

```

C READ (5,301) NET,NNT,NCN,NOFN,NMAT,NPROB
301 FORMAT (8I5)
NNE = 8

```

```

C DO 4 I = 1, NCN
4 READ (5,300) M,R(M),Z(M)
300 FORMAT (15,2F10.5)
K=NNE + 1
302 READ (5,302) ((NN(I,J),J=1,K),I=1,NET)
FORMAT (9I5)

```

```

C CALCULATION OF MID-NODE COORDINATES.

```

```

C DO 5 L = 1, NET
5 Z(NN(L,2)) = (R(NN(L,1))+R(NN(L,3)))/2.D0
R(NN(L,2)) = (Z(NN(L,1))+Z(NN(L,3)))/2.D0
Z(NN(L,4)) = (Z(NN(L,3))+R(NN(L,5)))/2.D0
R(NN(L,4)) = (Z(NN(L,3))+Z(NN(L,5)))/2.D0
Z(NN(L,6)) = (R(NN(L,5))+R(NN(L,7)))/2.D0
R(NN(L,6)) = (Z(NN(L,5))+Z(NN(L,7)))/2.D0
Z(NN(L,8)) = (R(NN(L,7))+R(NN(L,1)))/2.D0
R(NN(L,8)) = (Z(NN(L,7))+Z(NN(L,1)))/2.D0

```

```

C IF (NOFN.EQ.0) GO TO 7

```

```

C DO 6 I = 1, NOFN
6 READ (5,300) M,R(M),Z(M)

```

```

C 7 CONTINUE

```

```

C DO 19 I = 1, NNT
19 HCO(I) = Z(I)
VCO(I) = R(I)
CONTINUE

```

```

C CALL PLOTZR(HCO,VCO,NNT)

```

```

C

```

```

00002750
00002760
00002770
00002780
00002790
00002800
00002810
00002820
00002830
00002840
00002850
00002860
00002870
00002880
00002890
00002900
00002910
00002920
00002930
00002940
00002950
00002960
00002970
00002980
00002990
00003000
00003010
00003020
00003030
00003040
00003050
00003060
00003070
00003080
00003090
00003100
00003110
00003120
00003130
00003140
00003150
00003160
00003170
00003180
00003190
00003200
00003210
00003220

```



```

DO 1 I = 1, NMAT
1 READ (5,303) E(I),AL(I),POI(I),TK(I),DENS(I),SHT(I)
303 FORMAT (2D12.4,4F8.3)
C
C
C
C ARE THE UNITS IN METRIC OR BRITISH UNIT SYSTEM?
C
C READ (5,310) Q1
310 FORMAT (A4)
C
C WRITE (6,200)
200 FORMAT (1H1,45X,'** TRANSIENT THERMAL STRESS PROBLEM',
1, '**',/,49X,32(' '),//)
C
C WRITE (6,201) NET,NNT,NCN,NMAT,NPROB
201 FORMAT (/,15X,'TOTAL NUMBER OF ELEMENTS...',15,
1/,15X,'TOTAL NUMBER OF NODES...',15,/,15X,
2,'TOTAL NUMBER OF CORNER NODES.',15,/,15X,'TOTAL',
3,'NUMBER OF MATERIALS...',15,/,15X,'TOTAL NUMBER',
4,'OF PROBLEMS...',15,//)
C
C WRITE (6,204)
204 FORMAT (/,15X,'ELEMENT NO.',8X,'GLOBAL NODE NUMBER',
132X,'MATERIAL NO.',/)
C
C DO 2 I=1, NET
2 WRITE (6,205) I,(NN(I,J),J=1,K)
205 FORMAT (15X,15,10X,9(15,2X))
C
C IF (Q1.NE.BRIT) GO TO 10
C
C WRITE (6,211)
WRITE (6,213)
GO TO 11
C
C 10 WRITE (6,212)
11 WRITE (6,215)
WRITE (6,202)
C
C 202 FORMAT (/,15X,'THE RADIAL AND LONGITUDINAL',
1,' COORDINATES',/,/,15X,'NODE NO.',14X,'**R**',
212X,'**Z**',/)

```


00003710
00003720
00003730
00003740
00003750
00003760
00003770
00003780
00003790
00003800
00003810
00003820
00003830
00003840
00003850
00003860
00003870
00003880
00003890
00003900
00003910
00003920
00003930
00003940
00003950
00003960
00003970
00003980
00003990
00004000
00004010
00004020
00004030
00004040
00004050
00004060
00004070
00004080
00004090
00004100
00004110
00004120
00004130
00004140
00004150
00004160
00004170
00004180

```

C      WRITE (6,216) (I,R(I),Z(I),I=1,NNI)
C      FORMAT (16X,I4,13X,F10.4,6X,F10.4)

C      211 FORMAT (//,15X,'** THE FOLLOWING UNITS ARE USED **',/,
115X,'NAME',16X,'SYMBOL',14X,'UNIT',/,15X,4(' '),16X,
215X,'**',14X,'**',/,15X,'TIME',16X,'TIME',DTI',16X,
312X,'SECOND',/,15X,'TEMPERATURE',9X,'TEMP',FT',11X,
41DEGREE FAHRENHEIT',/,15X,'VELOCITY',12X,'VEL',17X,
51FEET PER SECOND',/,15X,'COORDINATES',9X,'R,Z',19X,
617X,'INCHES',/,15X,'MOD. OF ELASTICITY',2X,'E',19X,
71POUND FORCE/SQ.INCHES',/,15X,'POISSON S RATIO',5X,
81POI',17X,'DIMENSIONLESS')

C      213 FORMAT (15X,'COEF. OF THER. EXP.',15X,'THER. CONDUCTIVITY',
13X,'AL',18X,'1/DEGREE F.',15X,'THER. CONDUCTIVITY',13X,
23X,'TK',18X,'BTU/FT.HR.',15X,'DENSITY',13X,
31DENS',16X,'LB/FT.CUBE',/,15X,'SPECIFIC HEAT',7X,
41SHT',17X,'BTU/LB.DEG.F.',15X,'HEAT TRANSFER COEF',
52X,'HTCO,HTCI',11X,'BTU/FT.SQ.HR.,DEG.F.',/)

C      212 FORMAT (//,15X,'THE FOLLOWING UNITS ARE BEING USED',
1//,15X,'NAME',16X,'SYMBOL',14X,'UNIT',/,15X,'**',DTI',
216X,'**',14X,'**',/,15X,'TIME',16X,'TIME',DTI',
312X,'SECOND',/,15X,'TEMPERATURE',9X,'TEMP',FT',11X,
41CELSIUS',/,15X,'VELOCITY',12X,'VEL',17X,'METER',
51SEC',/,15X,'COORDINATES',9X,'R,Z',17X,'CENTIMETER',
6//,15X,'MOD. OF ELASTICITY',3X,'E',19X,'KG./CM.SQ.',/,
715X,'COEF. OF THER. EXP.',3X,'AL',18X,'1/CELSIUS')

C      215 FORMAT (15X,'THER. CONDUCTIVITY',3X,'TK',18X,'CAL/HR.M.CEL',
11SIUS',/,15X,'DENSITY',13X,'DENS',16X,'GM./CM.CI',
21UBE',/,15X,'SPECIFIC HEAT TRANSFER COEF',17X,'CAL./KG.H',
41CELSIUS',/,15X,'HEAT TRANSFER COEF',2X,'HTCO,H',
51TCI',11X,'CAL/HR.M.SQ.CELSIUS',/)

C      206 WRITE (6,206)
C      FORMAT (//,15X,'* PROPERTIES OF THE MATERIALS *',/,
115X,'MATERIAL',5X,'MODULUS OF',5X,'POISSON S',5X,
21THERM.EXP',5X,'THERMAL',10X,'VOLUME',6X,'SPECIFIC')

C      WRITE (6,207)

```



```

00004190
00004200
00004210
00004220
00004230
00004240
00004250
00004260
00004270
00004280
00004290
00004300
00004310
00004320
00004330
00004340
00004350
00004360
00004370
00004380
00004390
00004400
00004410
00004420
00004430
00004440
00004450
00004460
00004470
00004480
00004490
00004500
00004510
00004520
00004530
00004540
00004550
00004560
00004570
00004580
00004590
00004600
00004610
00004620
00004630
00004640
00004650
00004660
00004670
00004680
00004690
00004700
00004710
00004720
00004730
00004740
00004750
00004760
00004770
00004780
00004790
00004800
00004810
00004820
00004830
00004840
00004850
00004860
00004870
00004880
00004890
00004900
00004910
00004920
00004930
00004940
00004950
00004960
00004970
00004980
00004990
00005000

207 FORMAT (15X,'NUMBER',7X,'ELASTICITY',5X,'RATIO',9X,
1,'COEFFICIENT',4X,'CONDUCTIVITY',5X,'DENSITY',5X,
2,'HEAT',//)
3 DO 3 I=1,NMAT
3 WRITE (6,209) I,E(I),POI(I),AL(I),TK(I),DENS(I),SHT(I)
209 FORMAT (14X,I5,6X,F10.1,6X,F9.5,5X,F10.7,6X,F8.4,9X,
1F8.4,5X,F8.4,/)
CONVERSION OF THE SYSTEM OF UNITS.
IF (Q1.EQ.BRIT) GO TO 60
TREF = 20.D0
DO 61 I = 1, NMAT
DENS(I) = DENS(I) / 1000.D0
61 E(I) = E(I) * 100.D0
GO TO 63
60 TREF = 68.D0
DO 62 I = 1, NMAT
DENS(I) = DENS(I) / 1728.D0
62 TK(I) = TK(I) / 14200.D0
63 CONTINUE
DETERMINATION OF THE HALF-BAND-WIDTH OF THE
SYSTEM STIFFNESS MATRIX
IDIF = 0
NNE1 = NNE - 1
DO 22 I = 1, NET
IDIF1 = 0
DO 21 J = 2, NNE1
IDIF2 = NN(I,1) - NN(I,J)
IF (IDIF2.LT.0) IDIF2 = -IDIF2
IF (IDIF2.GT.IDIF1) IDIF1 = IDIF2
21 CONTINUE
IF (IDIF1.GT.IDIF) IDIF = IDIF1
22 CONTINUE
NBAND = IDIF + 1
IBAND = 2 * NBAND

```


00004670
00004680
00004690
00004700
00004710
00004720
00004730
00004740
00004750
00004760

```
C      NDF = 2 * NNT  
      WRITE (6,220) IBAND, NDF  
220    FORMAT (/,'15X','HALF-BAND-WIDTH OF THE STIFFNESS MATRIX IS',  
118,/, '15X','NUMBER OF DEGREES OF FREEDOM OF THE SYSTEM IS',15,/)
C      RETURN  
      END  
C  
C
```


SUBROUTINE PROB

```

** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** 
** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** 
** ** ** ** **   IN THIS SUBROUTINE THE THERMAL
** ** ** **   INITIAL AND BOUNDARY CONDITIONS AND THE VARIOUS
** ** ** **   OTHER TYPES OF LOADING WITH SPECIFIED STRUCTURAL
** ** ** **   BOUNDARY CONDITIONS ARE BEING READ IN AND PRINTED
** ** ** **   OUT. THE STEP SIZE OF TIME INTEGRATION, THE INTERVAL
** ** ** **   OF TEMPERATURE PRINTS IF REQUIRED, AND THE NUMBER
** ** ** **   OF THERMAL LOAD VECTORS IF WANTED ARE READ IN HERE.
** ** ** ** ** 
** ** ** ** 
IMPLICIT REAL * 8 (A-H,O-Z)

COMMON /ONE/ AJ(2,2),B(4,16),COPY(37,298),D(4,4),DL(4,4),
1DNZR(2,8),ESM(16,16),F(24,298),SSM(298,66),STOR(20,150)

COMMON /TWO/ AL(5),DENS(5),DETA(8),DXI(8),E(5),EPSO(4),
1FE(16),GC(4),GX(4),PGI(5),R(149),SHP(8),SHI(5),TK(5),Z(149)

COMMON /TRE/ AXIALF,DTI,EVERY,FORS1,OMEGA,HTCI,HTCO,
1POSI,PRES,Q1,Q2,Q3,SHMAX,SMAX,TIME,TIME1,TINIT,TMAXM,TREF,
2TMAXO,RMAXM,RMAXO,ZMAXM,ZMAXO,Q4,BIG

COMMON /FOR/ NN(40,9)

COMMON /FIV/ NCFI(37),NCFO(37),NNR(37),NRE(37),NNL(37),
1NPN(37)
COMMON /SIX/ IBAND,IEXT,INTP,IVEC,JVEC,KK,LOAD,NCFIT,NCFOT,
1NCHECK,NET,NDF,NGP,NNE,NNLT,NNRT,NPASS,NPROB,NRPPI,NRPPO,
3NNT,NNRE,NPNT,IPLANE,NBAND

EQUIVALENCE (C,SSM), (Y,SSM(1,18)), (T,SSM(1,35)),
1(BTE,SSM(1,36)), (TEMP,SSM(1,37)), (V,SSM(1,38)),
2(FI1,SSM(1,39)), (FI2,SSM(1,40)), (CE,SSM(1,41)),
3(YE,SSM(1,42)), (OFF1,SSM(1,43)), (OFF2,SSM(1,44)),
4(DIAG,SSM(1,45)), (BDRY1,SSM(1,46)), (BDRYO,SSM(1,47)),
5(ENT11,SSM(1,48)), (ENT1O,SSM(1,49)), (ENTEI,SSM(1,50)),
6(ENTEO,SSM(1,51)), (VOL,SSM(1,52)), (VOLO,SSM(1,53))

DIMENSION C(149,33),Y(149,33),T(149),BTE(149),TEMP(149),
1V(149),FI1(90),FI2(90),CE(8,8),YE(8,8),OFF1(37),OFF2(37),

```



```

2DIAG(37),BDRYI(16,5),BDRYO(16,5),VOL(36),VOLO(36),ENTE0(75),
3ENTE1(75),ENTIO(75),ENTII(75)
C      DATA BRIT/'BRIT'/'
C      TIME = 0.00
C      JVEC = 0
C      SMAX = 0.00
C      SHMAX = 0.00
C      NCP = 4
C      GX(1) = -.861136311594053
C      GX(2) = -.GX(1)
C      GX(3) = .339981043584856
C      GX(4) = -.GX(3)
C      GC(1) = .347854845137454
C      GC(2) = .GC(1)
C      GC(3) = .652145154862546
C      GC(4) = .GC(3)
C      PAI = 3.141592653589793

```

```

NNRE IS THE TOTAL NUMBER OF NODES AT RIGHT HAND END.
NRE(I) IS THE ARRAY OF GLOBAL NODE NUMBER FOR THE
RIGHT HAND END.
IPLANE.GT.0 IF PLANES REMAIN PLANE AT THE ENDS.
IPLANE.EQ.0 IF ONE END IS FREE OR RADIALLY FIXED.
IEXT IS AN INDICATION IF TEMPERATURE VECTORS ARE
GENERATED OR BEING READ IN AS FOLLOWS.
IEXT.EQ.0 TEMPERATURE VECTORS ARE GENERATED.
IEXT.GT.0 TEMPERATURE VECTORS ARE INPUTED.

```

```

READ (5,101)OMEGA,PRES,IEXT,LOAD,NNLT,NNRT,IPLANE
IF (OMEGA.EQ.0.00) GO TO 2
JVEC = JVEC + 1
WRITE (6,203) OMEGA
2 N = 0
IF (PRES.EQ.0.00) GO TO 6
READ (5,102) NPNT,(NPN(I),I=1,NPNT)

```

```

NPNT IS THE TOTAL NUMBER OF PRESSURE NODES
NPN(I) IS THE GLOBAL NODE NUMBER OF THE PRESSURE NODE I .
WRITE (6,444) NPNT,(NPN(I),I=1,NPNT)

```

```

00005230
00005240
00005250
00005260
00005270
00005280
00005290
00005300
00005310
00005320
00005330
00005340
00005350
00005360
00005370
00005380
00005390
00005400
00005410
00005420
00005430
00005440
00005450
00005460
00005470
00005480
00005490
00005500
00005510
00005520
00005530
00005540
00005550
00005560
00005570
00005580
00005590
00005600
00005610
00005620
00005630
00005640
00005650
00005660
00005670
00005680
00005690
00005700

```


00006190
00006200
00006210
00006220
00006230
00006240
00006250
00006260
00006270
00006280
00006290
00006300
00006310
00006320
00006330
00006340
00006350
00006360
00006370
00006380
00006390
00006400
00006410
00006420
00006430
00006440
00006450
00006460
00006470
00006480
00006490
00006500
00006510
00006520
00006530
00006540
00006550
00006560
00006570
00006580
00006590
00006600
00006610
00006620
00006630
00006640
00006650
00006660

IF CONVECTION BDRY. INSIDE Q3 > 0.00
IF INSULATED BDRY. INSIDE Q3 = 0.00

Q4.GT.0.00 TEMPERATURE PROBLEM IS TO BE SOLVED ONLY
Q4=0.00 STRESS PROBLEM IS TO BE SOLVED ALSO.
Q4.LT.0.00 STRESS PROBLEM IS TO BE SOLVED ONLY.

IF (Q2.GT. 0.00) GO TO 1
GO TO 3
1 CONTINUE

READ (5,103) HTCO , TEMPO , NCFOT , NRAMPO
N = NCFOT
READ (5,102) (NCFO(I), I = 1 , NCFOT)
NRPPO = NRAMPO + 1

READ (5,104) ((BDRYO(I,J),J=1,3),BDRYO(I,5),I = 2,NRPPO)

BDRYO(I,1) IS THE INITIAL TIME, TO BE READ IN FOR
EACH RAMP SEGMENT.
BDRYO(I,2) IS THE INITIAL TEMPERATURE, TO BE READ IN.
BDRYO(I,3) IS THE FINAL TEMPERATURE TO BE READ IN IF IT DIFFERS
FROM THE NEXT INITIAL TEMPERATURE,
BDRYO(I,4) IS THE SLOPE, WHICH IS CALCULATED BY THE
PROGRAM.
BDRYO(I,5) IS THE VELOCITY OF THE FLUID TO BE READ IN
IF IT DIFFERS FROM THE PREVIOUS RAMP VELOCITY

WRITE (6,201) HTCO

DO 4 I = 1, NCFOT
FT2(I) = TEMPO
4 FT1(I) = TEMPO

WRITE (6,201) (NCFO(I),FT1(I),I=1,NCFOT)

BDRYO(1,1) = 0.00
BDRYO(1,2) = TEMPO
BDRYO(1,4) = 0.00


```

C      BDRYO(1,5) = 0.00
      BDRYO(NRPP0,4) = 0.00
      BDRYO(NRPP0+1,1) = 999999.00

      DO 5 I = 2, NRAMPO
      IF (BDRYO(I,3).EQ.0.00) BDRYO(I,3) = BDRYO(I+1,2)
      IF (BDRYO(I,5).EQ.0.00) BDRYO(I,5) = BDRYO(I-1,5)
      DUM = BDRYO(I,3) - BDRYO(I,2)
      5 BDRYC(I,4) = DUM/(BDRYO(I+1,1) - BDRYO(I,1))

C
C      WRITE (6,202) (BDRYO(I,1),BDRYO(I,2),BDRYO(I+1,1),
1 BDRYO(I,3),BDRYO(I,4),BDRYO(I,5),I = 2,NRPP0)

C      IF (Q1.EQ.BRIT) GO TO 20
      DO 21 I = 2, NRPP0
      21 BDRYO(I,5) = BDRYC(I,5) * 100.00
      GO TO 18
      DO 22 I = 2, NRPP0
      22 BDRYO(I,5) = BDRYC(I,5) * 12.00

C      GO TO 18

C
C      3 WRITE (6,1000)
      18 CONTINUE

C      IF (Q3.GT.000) GO TO 7
      WRITE (6,1001)
      GO TO 9

C      7 READ (5,103) HTCI, TEMPI, NCFIT, NRAMPI
      READ (5,102) (NCFI(I), I = 1, NCFIT)

C      NRPPI = NRAMPI + 1
      READ (5,104) (BDRYI(I,J),J=1,3),BDRYI(I,5),I=2,NRPP1)

C
C      WRITE (6,202) HTCI

C
C      DO 10 I = 1, NCFIT
      K = N + I
      FI1(K) = TEMPI
      10 FT2(K) = TEMPI

C
C      DO 19 I = 1, NCFIT

```



```

C C
K = N + I
19 WRITE (6,201) NCFI(I) , FT1(K)
C C
      BDRYI(1,1) = 0.00
      BDRYI(1,2) = TEMPI
      BDRYI(1,4) = 0.00
      BDRYI(1,5) = 0.00
      BDRYI(NRPP1,4) = 0.00
      BDRYI(NRPP1+1,1) = 999999.00
C C C
      DO 11 I = 2, NRAMPI
      IF (BDRYI(I,3).EQ.0.00) BDRYI(I,3) = BDRYI(I+1,2)
      IF (BDRYI(I,5).EQ.0.00) BDRYI(I,5) = BDRYI(I-1,5)
      DUM = BDRYI(I,3) - BDRYI(I,2)
      11 BDRYI(I,4) = DUM / (BDRYI(I+1,1)-BDRYI(I,1))
C
      WRITE (6,202) (BDRYI(I,1),BDRYI(I,2),BDRYI(I+1,1),
      1BDRYI(I,3),BDRYI(I,4),BDRYI(I,5),I=2,NRPP1)
C C
      IF (Q1.EQ.BRIT) GO TO 23
      DO 24 I = 2, NRPP1
      24 BDRYI(I,5) = BDRYI(I,5) * 100.00
      GO TO 9
      DO 25 I = 2, NRPP1
      25 BDRYI(I,5) = BDRYI(I,5) * 12.00
C
      9 READ (5,106) DTI , TIME1 , EVERY , IVEC , INTP
C C C
      TS = TIME1 + EVERY * (IVEC-1)
C
      WRITE (6,215) DTI
      215 FORMAT (15X,'STEP SIZE OF TIME INTEGRATION IS ',F10.4,/)
C
      DO 14 I = 1,NNT
      TEMPI(I) = TINIT
      WRITE (6,204) TINIT
      WRITE (6,205) TREF
      WRITE (6,206) IVEC
C
      WRITE (6,208) TIME1
      WRITE (6,209) EVERY
      IF (INTP.EQ.0) GO TO 40
      WRITE (6,210) INTP
      GO TO 41

```

```

00007150
00007160
00007170
00007180
00007190
00007200
00007210
00007220
00007230
00007240
00007250
00007260
00007270
00007280
00007290
00007300
00007310
00007320
00007330
00007340
00007350
00007360
00007370
00007380
00007390
00007400
00007410
00007420
00007430
00007440
00007450
00007460
00007470
00007480
00007490
00007500
00007510
00007520
00007530
00007540
00007550
00007560
00007570
00007580
00007590
00007600
00007610
00007620

```



```

40 WRITE (6,211)
41 CONTINUE
C 208 FORMAT (15X,'THE FIRST TEMP. VECTOR IS KEPT AT TIME ',
1F10.4,/)
C 209 FORMAT (15X,'THEN TEMP. VECTORS ARE KEPT AT EVERY',
13X,F10.4,' SECONDS',/)
C 210 FORMAT (15X,'TEMPERATURES ARE PRINTED FOR EVERY',5X,15,
1,' INTERVAL OF TIME',/)
C 211 FORMAT (15X,'TEMPERATURES ARE NOT PRINTED',/)

KK = 1
NPASS = TS / DTI
NPASS = NPASS + NPASS/10

C 201 FORMAT (15X,'NODE',15,2X,'HAS CONTACT WITH FLUID AT',2X,F10.4,
13X,'DEGREE TEMPERATURE')
C 202 FORMAT (//,15X,'ENTRY TEMPERATURE AND TIME RELATION',
1/,15X,34(' '),//,15X,'INITIAL TIME',5X,'INITIAL ',
2,'TEMPERATURE',10X,'FINAL TIME',5X,'FINAL TEMPERATURE',
310X,'SLOPE',5X,'VELOCITY',//,(17X,F10.4,10X,F10.4,
414X,F10.2,8X,F10.4,10X,F10.4,5X,F10.4))
C 203 FORMAT (//,15X,'THE SYSTEM IS ROTATING ',F10.5,' REV',
1,'OLUTIONS PER MINUTE',/)
C 204 FORMAT (//,15X,'INITIAL TEMPERATURE OF THE SOLID IS',
1,' ',F10.3,/)
C 205 FORMAT (15X,'REFERENCE TEMPERATURE IS',12X,F10.3,/)
C 206 FORMAT (15X,'NUMBER OF STORED TEMPERATURE VECTORS IS',16,/)
C 207 FORMAT (15X,'INSIDE PRESSURE IS',19X,F10.5,/)
C 1000 FORMAT (//,15X,'OUTSIDE BDY. COND. IS INSULATED',/)
C 1001 FORMAT (//,15X,'INSIDE BDY. COND. IS INSULATED',/)
C 2021 FORMAT (//,15X,'OUTSIDE BDY. COND. IS CONVECTION',//,

```



```

115X, 'AND THE HEAT TRANSFER COEF. IS ', 1PD14.6, '/')
2022 FORMAT (//, 15X, 'INSIDE BDRY. COND. IS CONVECTION', //,
115X, 'AND THE HEAT TRANSFER COEF. IS ', 1PD14.6, '/')
C
C
52 IF (IEXT.EQ.0) GO TO 54
   NNT1 = NNT + 1
   READ (5, 109) ((STOR(I, J), J=1, NNT1), I=1, IEXT)
   IVEC = IEXT
   WRITE (6, 212) IEXT
212 FORMAT (//, 15X, 'TEMPERATURE VECTORS ARE READ IN AND',
1//, 15X, 'TEMPERATURES ARE IN ORDER OF GLOBAL NODE',
2, 'NUMBERS : ', //)
C
DO 56 I = 1, IEXT
DO 56 J = 1, NNT
WRITE (6, 213) J, STOR(I, J), F10.3)
213 FORMAT (15X, 'NODE(', I3, ') ', F10.3)
54 CONTINUE
IF (Q1.EQ.BRIT) GO TO 60
GO TO 62
C
60 IF (Q2.GT.0.D0) HTCO = HTCO / 518400.D0
   IF (Q3.GT.0.D0) HTCI = HTCI / 518400.D0
C
GO TO 58
C
57 IVEC = 1
STOR(1, NNT+1) = 0.D0
DO 59 J = 1, NNT
STOR(1, J) = TREF
59 CONTINUE
62 OMEGA = OMEGA * (PAI / 30.D0)
C
BIG = 10.D0**20
100 FORMAT (5F10.4)
101 FORMAT (2F10.4, 5I5)
C
102 FORMAT (12I5)
103 FORMAT (D16.4, F10.4, 2I5)
C
104 FORMAT (4F10.4)
106 FORMAT (3F10.4, 2I5)
C
108 FORMAT (6F10.4)
109 FORMAT (6F10.3)
C
RETURN
END
00008110
00008120
00008130
00008140
00008150
00008160
00008170
00008180
00008190
00008200
00008210
00008220
00008230
00008240
00008250
00008260
00008270
00008280
00008290
00008300
00008310
00008320
00008330
00008340
00008350
00008360
00008370
00008380
00008390
00008400
00008410
00008420
00008430
00008440
00008450
00008460
00008470
00008480
00008490
00008500
00008510
00008520
00008530
00008540
00008550
00008560
00008570
00008580
00008590
00008600

```



```

00008610
00008620
00008630
00008640
00008650
00008660
00008670
00008680
00008690
00008700
00008710
00008720
00008730
00008740
00008750
00008760
00008770
00008780
00008790
00008800
00008810
00008820
00008830
00008840
00008850
00008860
00008870
00008880
00008890
00008900
00008910
00008920
00008930
00008940
00008950
00008960
00008970
00008980
00008990
00009000
00009010
00009020
00009030
00009040
00009050
00009060

SUBROUTINE PLOTZR(X , Y , N)
*****
SUBROUTINE PLOTZR WILL PRODUCE A PLOT OF THE NODAL
POINTS OF ANY TWO DIMENSIONAL GEOMETRY. THE NODES
ARE REPRESENTED BY *, AND THE APPROPRIATE NODAL
POINT NUMBERS ARE PRINTED BELOW EACH NODE.
PLOTZR(X,Y,N) ; X IS HORIZONTAL COORDINATE
Y IS VERTICAL COORDINATE
N IS NUMBER OF NODES. (LESS THAN 1000)
*****
DIMENSION X(N),Y(N),IDUM(104),JDUM(104),NU(10)
DATA NU/1,2,3,4,5,6,7,8,9,10./
DATA ISTAR/'*'/ , IBLANK/' ' , IDOT/'.'/
DO 20 I = 1, 104
20 IDUM(I) = IDOT
WRITE (6,85) IDUM
85 FORMAT (1H1,/,45X,'GRAPHICAL PRESENTATION OF NODAL',
1,POINT S.,/,48X,'Z HORIZONTAL R VERTICAL',/,20X,
2104A1)
XMAX = -1.0E 15
YMAX = -1.0E 15
XMIN = - XMAX
YMIN = - YMAX
DO 1 I = 1, N
1 IF (X(I).GT.XMAX) XMAX = X(I)
IF (X(I).LT.XMIN) XMIN = X(I)
IF (Y(I).GT.YMAX) YMAX = Y(I)
IF (Y(I).LT.YMIN) YMIN = Y(I)
1 CONTINUE
DIFX = XMAX - XMIN

```



```

C      DIFY = YMAX - YMIN
      FACTX = 100.0 / DIFY
      FACTY = 70.0 / DIFY

      DO 2 I = 1, N
      X(I) = FACTX * (X(I) - XMIN)
      Y(I) = 1.0 + FACTY * (Y(I) - YMIN)
2 CONTINUE

C      WRITE (6,87) YMAX
C
C      JDUM(1) = IDOT
      IDUM(103) = IBLANK
      K = 71
      DO 9 I = 1, 71
      JDUM(104) = IDOT
      DO 6 J = 2, 103
      JDUM(J) = IBLANK
6 IDUM(J) = IBLANK
C
C      DO 7 J = 1, N
      M = Y(J)
      IF (M.EQ.K) GO TO 8
      GO TO 7
8 NUM = X(J) + 2
      IDUM(NUM) = ISTAR
C
C      K1 = J/100
      IF (K1.NE.0) GO TO 15
      K1 = J/10
      IF (K1.NE.0) GO TO 10
      JDUM(NUM) = NU(J)
      GO TO 7
C      10 K2 = J - 10*K1
      JDUM(NUM) = NU(K1)
      IF (K2.EQ.0) K2 = 10
      JDUM(NUM+1) = NU(K2)
      GO TO 7
C      15 K3 = J - 100*K1
      K2 = K3 / 10
      K3 = K3 - 10*K2
      JDUM(NUM) = NU(K1)
      IF (K2.EQ.0) K2 = 10
      JDUM(NUM+1) = NU(K2)

```


00009550
00009560
00009570
00009580
00009590
00009600
00009610
00009620
00009630
00009640
00009650
00009660
00009670
00009680
00009690
00009700
00009710
00009720
00009730
00009740
00009750
00009760
00009770
00009780
00009790
00009800
00009810
00009820
00009830
00009840
00009850
00009860
00009870
00009880
00009890

```

C      IF (K3.EQ.0) K3 = 10
C      JDUM(NUM+2) = NU(K3)
C
C      7 CONTINUE
C
C      WRITE (6,88) IDUM
C      WRITE (6,88) JDUM
C      K = K - 1
C      9 CONTINUE
C
C      WRITE (6,89) YMIN
C
C      DIF = (XMAX-XMIN) / 4.0
C      X2 = XMIN + DIF
C      X3 = X2 + DIF
C      X4 = X3 + DIF
C      WRITE (6,90) XMIN , X2 , X3 , X4 , XMAX
C
C      87 FORMAT (3X,'MAX.R=',F10.5)
C      89 FORMAT (3X,'MIN.R=',F10.5)
C      88 FORMAT (20X,104A1)
C      90 FORMAT (20X,'+',24(' '),'+',24(' '),'+',
C      124(' '),'+...',/,16X,4(F10.5,15X),F10.5,'Z')
C
C      WRITE (6,555)
C      555 FORMAT (//,40X,'NOTE :',/,45X,'IF THE NODES ARE ',
C      1,'TOO CLOSED TO EACH OTHER',/,45X,'A NODE MAY NOT BE ',
C      2,'PRINTED',/,45X,'MAY BE MISPRINTED',/,
C      3,'OR BOTH.')
C      RETURN
C      END

```



```

C      00009900
C
C      SUBROUTINE PLOT(X,Y,LL)
C      DIMENSION X(75),Y(75),IDUM(102)
C      DATA ISTAR/'*'/ , IPLUS/'+'/ , IBLANK/' ' , IDOT/'.'/
C
C      *****
C      SUBROUTINE PLOT(X,Y,LL) WILL PLOT THE VARIATION OF
C      THE ENTRY TEMPERATURE VERSUS TIME AS 75 POINTS FOR
C      THE FIRST 150 TIME INCREMENTS, IF (TEMPERATURE.GE.O.)
C      X IS TIME.
C      Y IS TEMPERATURE.
C      LL=0 OUTSIDE FLOW.
C      LL=1 INSIDE FLOW.
C      *****
C
C      DO 1 I = 1, 75
C      IF (Y(I).LT.O.DD) GO TO 50
C      1 CONTINUE
C      DO 5 I = 1, 102
C      IDUM(I) = IDOT
C
C      YMAX = -1.OE 18
C      YMIN = -YMAX
C      DO 2 I = 1, 75
C      IF (Y(I).GT.YMAX) YMAX = Y(I)
C      IF (Y(I).LT.YMIN) YMIN = Y(I)
C      2 CONTINUE
C
C      RAT = 1.O + (100.O * YMIN) / YMAX
C      NUM = RAT
C      IDUM(NUM) = IPLUS
C
C      IF (LL.EQ.O) WRITE (6,66) IDUM
C      66 FORMAT (1H1, / , 42X, 'OUTSIDE FLUID ENTRY TIME TEMPERA',
C      1, 'TURE PLOT', / , 42X, 41('*'), / , 7X, 'TIME', 5X, 'TEMPERATURE',
C      290X, 'TEMPERATURE', / , 29X, 102A1)
C
C      00009910
C      00009920
C      00009930
C      00009940
C      00009950
C      00009960
C      00009970
C      00009980
C      00009990
C      00010000
C      00010010
C      00010020
C      00010030
C      00010040
C      00010050
C      00010060
C      00010070
C      00010080
C      00010090
C      00010100
C      00010110
C      00010120
C      00010130
C      00010140
C      00010150
C      00010160
C      00010170
C      00010180
C      00010190
C      00010200
C      00010210
C      00010220
C      00010230
C      00010240
C      00010250
C      00010260
C      00010270
C      00010280
C      00010290
C      00010300
C      00010310
C      00010320
C      00010330
C      00010340
C      00010350

```


SUBROUTINE LLT (N,A,WL)

THIS SUBROUTINE WILL DECOMPOSE ANY N BY N REAL
SYMMETRIC MATRIX A INTO A LOWER TRIANGULAR MATRIX
L, WL, OF THE SAME ORDER N, SUCH THAT THE PRODUCT
OF L, WL, AND ITS TRANSPOSE, IS THE GIVEN MATRIX A.
(THIS IS CHOLESKY DECOMPOSITION.)

```

IMPLICIT REAL *8 (A-H,O-Z)
DIMENSION A(N,N),WL(N,N)
DO 7 I=1,N
DO 8 J=1,N
WL(I,J)=0.0D0
DO CONTINUE
WL(I,1)=DSQRT(A(1,1))
DO 1 I=2,N
WL(I,1)=A(I,1)/WL(1,1)
N1=N-1
M=3
DO 2 I=2,N
I1=I-1
DO 3 J=1,I1
WL(I,I1)=WL(I,I)+WL(I,J)**2
WL(I,I1)=WL(I,I1)-WL(I,J)
WL(I,I1)=DSQRT(WL(I,I1))
IF (M.GT.N) GO TO 9
I1=M,I
DO 5 I1=M,N
L=1,I1
DO 6 L=1,I1
WL(I,I1)=WL(I,I1)+WL(I,L)*W
WL(I,I1)=WL(I,I1)-WL(I,L)
M=M+1
END

```

UUUUUUUUUUUUUUUUUUUUUUUUUUUU


```

C      0001 1020
SUBROUTINE CANDY
C      0001 1030
C      0001 1040
C      0001 1050
C      0001 1060
C      0001 1070
C      0001 1080
C      0001 1090
C      0001 1100
C      0001 1110
C      0001 1120
C      0001 1130
C      0001 1140
C      0001 1150
C      0001 1160
C      0001 1170
C      0001 1180
C      0001 1190
C      0001 1200
C      0001 1210
C      0001 1220
C      0001 1230
C      0001 1240
C      0001 1250
C      0001 1260
C      0001 1270
C      0001 1280
C      0001 1290
C      0001 1300
C      0001 1310
C      0001 1320
C      0001 1330
C      0001 1340
C      0001 1350
C      0001 1360
C      0001 1370
C      0001 1380
C      0001 1390
C      0001 1400
C      0001 1410
C      0001 1420
C      0001 1430
C      0001 1440
C      0001 1450
C      0001 1460
C      0001 1470

*****
SUBROUTINE CANDY WILL CALCULATE CAPACITANCE MATRIX
C AND ADMITTANCE MATRIX Y IN THE BANDED FORM THEN
THE ADDITIONAL YSTAR MATRIX IS EVALUATED AND ADDED
TO THE Y MATRIX. A AND G MATRICES ARE EVALUATED
AND PLACED IN THE C AND Y RESPECTIVELY.
FINALLY, A CHOLESKY DECOMPOSITION IS PERFORMED ON C.
*****
IMPLICIT REAL * 8 (A-H,O-Z)
COMMON /ONE/ AJ(2,2),B(4,16),COPY(37,298),D(4,4),DL(4,4),
1DNZR(2,8),ESM(16,16),F(24,298),SSM(298,66),STOR(20,150)
COMMON /TWO/ AL(5),DENS(5),DETA(8),DXI(8),E(5),EPSO(4),
1FE(16),GC(4),GX(4),POI(5),R(149),SHP(8),SHI(5),TK(5),Z(149)
COMMON /TRE/ AXIALF,DTI,EVERY,FORS1,OMEGA,HTCI,HTCO,
1POSI,PRES,Q1,Q2,Q3,SHMAX,SMAX,TIME,TIME1,TINIT,TMAXM,TREF,
2TMAXO,RMAXM,RMAXO,ZMAXM,ZMAXO,Q4,BIG
COMMON /FOR/ NN(40,9)
COMMON /FIV/ NCFI(37),NCFQ(37),NNR(37),NRE(37),NNL(37),
1NPN(37)
COMMON /SIX/ IBAND,IEXT,INTP,IVEC,JVEC,KK,LOAD,NCFIT,NCFOT,
1NCHECK,NET,NDF,NGP,NNIE,NNLT,NNRT,NPASS,NPROB,NRPPI,NRPPO,
3NNT,NNRE,NPNT,IPLANE,NBAND
EQUIVALENCE (C,SSM), (Y,SSM(1,18)), (T,SSM(1,35)),
1(BTE,SSM(1,36)), (TEMP,SSM(1,37)), (V,SSM(1,38)),
2(FT1,SSM(1,39)), (FT2,SSM(1,40)), (CE,SSM(1,41)),
3(YE,SSM(1,42)), (OFF1,SSM(1,43)), (OFF2,SSM(1,44)),
4(DIAG,SSM(1,45)), (RDRYI,SSM(1,46)), (BDRYO,SSM(1,47)),
5(ENTII,SSM(1,48)), (ENTIO,SSM(1,49)), (ENTEI,SSM(1,50)),
6(ENTEO,SSM(1,51)), (VOL,SSM(1,52)), (VLO,SSM(1,53))
DIMENSION C(149,33),Y(149,33),T(149),BTE(149),TEMP(149),
1V(149),FT1(90),FT2(90),CE(8,8),YE(8,8),OFF1(37),OFF2(37),

```


00011960
00011970
00011980
00011990
00012000
00012010
00012020
00012030
00012040
00012050
00012060
00012070
00012080
00012090
00012100
00012110
00012120
00012130
00012140
00012150
00012160
00012170
00012180
00012190
00012200
00012210
00012220
00012230
00012240
00012250
00012260
00012270
00012280
00012290
00012300
00012310
00012320
00012330
00012340
00012350
00012360
00012370
00012380
00012390
00012400
00012410
00012420
00012430

DERIVATIVES OF THE SHAPE FUNCTIONS WITH RESPECT TO XI.

```

DXI(1)      = .25D0 * (1.D0-ETA) * (2.D0*XI+ETA)
DXI(2)      = -XI * (1.D0-ETA)
DXI(3)      = .25D0 * (1.D0-ETA) * (2.D0*XI-ETA)
DXI(4)      = .5D0 * (1.D0-ETA**2)
DXI(5)      = .25D0 * (1.D0+ETA) * (2.D0*XI+ETA)
DXI(6)      = -XI * (1.D0+ETA)
DXI(7)      = .25D0 * (1.D0+ETA) * (2.D0*XI-ETA)
DXI(8)      = -.5D0 * (1.D0-ETA**2)

```

DERIVATIVES OF THE SHAPE FUNCTIONS WITH RESPECT TO ETA.

```

DETA(1)     = .25D0 * (1.D0-XI) * (2.D0*ETA+XI)
DETA(2)     = -.5D0 * (1.D0-XI**2)
DETA(3)     = .25D0 * (1.D0+XI) * (2.D0*ETA-XI)
DETA(4)     = -ETA * (1.D0+XI)
DETA(5)     = .25D0 * (1.D0+XI) * (2.D0*ETA+XI)
DETA(6)     = .5D0 * (1.D0-XI**2)
DETA(7)     = .25D0 * (1.D0-XI) * (2.D0*ETA-XI)
DETA(8)     = -ETA * (1.D0-XI)

```

CONSTRUCTION OF THE JACOBIAN MATRIX.

```

DO 5 I = 1, 2
DO 5 J = 1, 2
5 AJ(I,J) = 0.D0

```

```

DO 22 I = 1, NNE
AJ(1,1) = AJ(1,1) + Z(NN(L,I))
AJ(1,2) = AJ(1,2) + DXI(1) * Z(NN(L,I))
AJ(2,1) = AJ(2,1) + DETA(1) * Z(NN(L,I))
AJ(2,2) = AJ(2,2) + DETA(1) * Z(NN(L,I))
22 CONTINUE

```

```

DET = AJ(1,1) * AJ(2,2) - AJ(1,2) * AJ(2,1)
DEI = 1.D0 / DET

```

CONSTRUCTION OF INVERSE OF JACOBIAN

C C

```

DUM = AJ(1,1) * DEI
AJ(1,1) = AJ(2,2) * DEI
AJ(1,2) = -AJ(1,2) * DEI
AJ(2,1) = -AJ(2,1) * DEI
AJ(2,2) = DUM

```

C

```

CR = 0.0
DO 21 I = 1, NNE
  M1 = NN(L,I)
  CR = CR + SHP(I) * R(M1)

```

21

C C C C C

EVALUATION OF THE UPPER TRIANGLE OF CE.

```

DUM = FACTC * DET * CR * GC(N) * GC(M)
DO 23 I = 1, NNE
  DO 23 J = I, NNE
    CE(I,J) = CE(I,J) + DUM * SHP(I) * SHP(J)
  CE(I,J) = CE(I,J) + DUM * SHP(I) * SHP(J)

```

23

C C C C C C C

CONSTRUCTION OF DNZR ; THE MATRIX OF THE PARTIAL
DERIVATIVES OF SHAPE FUNCTIONS RESPECT TO Z AND R

```

DO 30 I = 1, NNE
  DNZR(1,I) = AJ(1,1) * DXI(I) + AJ(1,2) * DETA(I)
  DNZR(2,I) = AJ(2,1) * DXI(I) + AJ(2,2) * DETA(I)

```

30

C C C C C

EVALUATION OF UPPER TRIANGLE OF YE.

```

DUM = FACTY * CR * DET * GC(N) * GC(M)
DO 26 I = 1, NNE
  DO 26 J = I, NNE
    DUM1 = DNZR(1,I) * DNZR(1,J)
    DUM2 = DNZR(2,I) * DNZR(2,J)
    YE(I,J) = YE(I,J) + (DUM1 + DUM2) * DUM

```

26

C C C

9 CONTINUE


```

C      DO 27 I = 1, NNE
      DO 27 J = 1, NNE
      CE(J,I) = CE(I,J)
      27 YE(J,I) = YE(I,J)
C
C      GENERATING THE THERMAL CAPACITANCE AND ADMITTANCE
C      MATRICES IN THE BANDED FORM.
C
      14 DO 14 I = 1, NNE
      II = NN(L,I) - 1, NNE
      DO 14 J = 1, NNE
      JJ = NN(L,J) - 1, NNE
      IF (II.GT.JJ) GO TO 14
      JJ = JJ - II
      JJ = JJ + 1
      KK = JJ + 1
      C(JJ,KK) = C(JJ,KK) + CE(I,J)
      Y(JJ,KK) = Y(JJ,KK) + YE(I,J)
      14 CONTINUE
C
      25 CONTINUE
C
      IF (Q2.GT.0.D0) GO TO 40
      NC = 0
      GO TO 50
      40 NC = NCFOT
      DO 64 I = 1, NCFOT
      DIAG(I) = 0.D0
      OFF1(I) = 0.D0
      64 OFF2(I) = 0.D0
C
      NECFO = (NCFOT-1) / 2
      DO 60 M = 1, NECFO
      K = 2 * M - 1
      JL = NCFOT(K)
      JM = NCFOT(K+1)
      JR = NCFOT(K+2)
      DO 60 N = 1, NGP
      XI = GX(N)

```

```

00012920
00012930
00012940
00012950
00012960
00012970
00012980
00012990
00013000
00013010
00013020
00013030
00013040
00013050
00013060
00013070
00013080
00013090
00013100
00013110
00013120
00013130
00013140
00013150
00013160
00013170
00013180
00013190
00013200
00013210
00013220
00013230
00013240
00013250
00013260
00013270
00013280
00013290
00013300
00013310
00013320
00013330
00013340
00013350
00013360
00013370
00013380
00013390

```



```

00013400
00013410
00013420
00013430
00013440
00013450
00013460
00013470
00013480
00013490
00013500
00013510
00013520
00013530
00013540
00013550
00013560
00013570
00013580
00013590
00013600
00013610
00013620
00013630
00013640
00013650
00013660
00013670
00013680
00013690
00013700
00013710
00013720
00013730
00013740
00013750
00013760
00013770
00013780
00013790
00013800
00013810
00013820
00013830
00013840
00013850
00013860
00013870

```

```

C      SHP1      = -.5D0 * (1.D0-XI) * XI
C      SHP2      = 1.D0 - XI**2
C      SHP3      = .5D0 * (1.D0+XI) * XI

C      DXI1      = -.5D0 + XI
C      DXI2      = -2.D0 * XI
C      DXI3      = .5D0 + XI

C      RR = SHP3*R(JL) + SHP2*R(JM) + SHP1*R(JR)

C      DRXI = DXI3*R(JL) + DXI2*R(JM) + DXI1*R(JR)
C      DZXI = DXI3*Z(JL) + DXI2*Z(JM) + DXI1*Z(JR)

C      DRXI**2
C      DZXI**2

C      DUMY = RR * GC(N) * DSQRT(DRXI+DZXI)

C      DIAG( K ) = DIAG( K ) + SHP3*SHP3 * DUMY
C      DIAG(K+1) = DIAG(K+1) + SHP2*SHP2 * DUMY
C      DIAG(K+2) = DIAG(K+2) + SHP1*SHP1 * DUMY
C      OFF1( K ) = OFF1( K ) + SHP3*SHP2 * DUMY
C      OFF1(K+1) = OFF1(K+1) + SHP2*SHP1 * DUMY
C      OFF2( K ) = OFF2( K ) + SHP1*SHP3 * DUMY

C      60 CONTINUE

C      CONST = 2.D0 * PAI * HTCO

C      DO 67 I = 1, NCFOT
C      DIAG(I) = DIAG(I) * CONST
C      OFF1(I) = OFF1(I) * CONST
C      67 OFF2(I) = OFF2(I) * CONST

C      ADDING THE CONTRIBUTION OF Y STAR TO Y MATRIX
C      FOR OUTSIDE CONVECTION

C      DO 68 I = 1, NCFOT
C      K = NCFOT(I)
C      68 Y(K,I) = Y(K,I) + DIAG(I)

C      N1 = NCFOT - 1
C      DO 69 I = 1, N1

```



```

00013880
00013890
00013900
00013910
00013920
00013930
00013940
00013950
00013960
00013970
00013980
00013990
00014000
00014010
00014020
00014030
00014040
00014050
00014060
00014070
00014080
00014090
00014100
00014110
00014120
00014130
00014140
00014150
00014160
00014170
00014180
00014190
00014200
00014210
00014220
00014230
00014240
00014250
00014260
00014270
00014280
00014290
00014300
00014310
00014320
00014330
00014340
00014350

```

```

K1 = NCFO(I)
K2 = NCFO(I+1)
K = K2 - K1 + 1
IF (K.LE.0) K = - K + 2
K1 = MINO(K1,K2)
Y(K1,K) = Y(K1,K) + OFF1(I)
69 CONTINUE

```

C

```

N2 = NCFOT-2
DO 66 I = 1, N2, 2
K1 = NCFO(I)
K2 = NCFO(I+2)
K = K2 - K1 + 1
IF (K.LE.0) K = - K + 2
K1 = MINO(K1,K2)
Y(K1,K) = Y(K1,K) + OFF2(I)
66 CONTINUE

```

C
C
C

```

50 IF (Q3.GT.0.D0) GO TO 41
GO TO 83
41 DO 54 I = 1, NCFIT
J = NC + I
DIAG(J) = 0.D0
OFF1(J) = 0.D0
54 OFF2(J) = 0.D0

```

C
C

```

NECFI = (NCFIT - 1) / 2
DO 59 M = 1, NECFI
K = 2 * M - 1

```

C
C

```

JL = NCFI(K)
JM = NCFI(K+1)
JR = NCFI(K+2)

```

C

```

DO 59 N = 1, NGP

```

C

```

XI = GX(N)

```

C
C

```

SHP1 = -.5D0 * (1.D0 - XI) * XI
SHP2 = 1.D0 * XI**2
SHP3 = .5D0 * (1.D0 + XI) * XI
DXI1 = -.5D0 * XI
DXI2 = -2.D0 * XI

```



```

C      DXI3      = .5D0 + XI
C      RR      = SHP1*R(JL) + SHP2*R(JM) + SHP3*R(JR)
C      DRXI    = DXI1*R(JL) + DXI2*R(JM) + DXI3*R(JR)
C      DZXI    = DXI1*Z(JL) + DXI2*Z(JM) + DXI3*Z(JR)
C      DRXI    = DRXI**2
C      DZXI    = DZXI**2
C      DUMY    = RR * GC(N) * DSQRT(DRXI+DZXI)
C      I      = NC + K
C      DIAG(I) = DIAG(I) + SHP1*SHP1 * DUMY
C      DIAG(I+1) = DIAG(I+1) + SHP2*SHP2 * DUMY
C      DIAG(I+2) = DIAG(I+2) + SHP3*SHP3 * DUMY
C      OFF1(I) = OFF1(I) + SHP1*SHP2 * DUMY
C      OFF1(I+1) = OFF1(I+1) + SHP2*SHP3 * DUMY
C      OFF2(I) = OFF2(I) + SHP1*SHP3 * DUMY
C
C      59 CONTINUE
C
C      CONST = 2.D0 * PAI * HTC1
C
C      DO 57 J = 1, NCFIT
C      I = NC + J
C      DIAG(I) = DIAG(I) * CONST
C      OFF1(I) = OFF1(I) * CONST
C      OFF2(I) = OFF2(I) * CONST
C
C      57
C
C      ADDING THE CONTRIBUTION OF Y STAR TO Y MATRIX
C      FOR INSIDE CONVECTION
C
C      DO 58 I = 1, NCFIT
C      J = NC + I
C      K = NCFI(I)
C      Y(K,I) = Y(K,I) + DIAG(J)
C      MI = NCFIT - I
C
C      58
C
C      DO 55 I = 1, M1
C      J = NC + I
C      K1 = NCFI(I)
C      K2 = NCFI(I+1)

```



```

00014840
00014850
00014860
00014870
00014880
00014890
00014900
00014910
00014920
00014930
00014940
00014950
00014960
00014970
00014980
00014990
00015000
00015010
00015020
00015030
00015040
00015050
00015060
00015070
00015080
00015090
00015100
00015110
00015120
00015130
00015140
00015150
00015160
00015170
00015180
00015190
00015200
00015210
00015220
00015230
00015240
00015250
00015260
00015270
00015280
00015290
00015300
00015310

```

```

K = K2 - K1 + 1
IF (K.LE.O) K = - K + 2
K1 = MINO(K1,K2)
Y(K1,K) = Y(K1,K) + OFF1(J)
55 CONTINUE

```

C

```

M2 = NCFIT - 2
DO 56 I = 1, M2, 2
J = NC + I
K1 = NCFI(I)
K2 = NCFI(I+2)
K = K2 - K1 + 1
IF (K.LE.O) K = - K + 2
K1 = MINO(K1,K2)
Y(K1,K) = Y(K1,K) + OFF2(J)
56 CONTINUE

```

C

```

83 CONTINUE

```

CCCCCCCC

EVALUATION OF A AND G MATRICES

```

DUM = .5000 * DTI
DO 1 I=1,NNT
DO 1 J=1,NBAND
EL = DUM * Y(I,J)
C1 = C(I,J) + EL
C2 = C(I,J) - EL
C(I,J) = C1
Y(I,J) = C2
1 CONTINUE

```

1

CCCCCCCCCCCCCCCC

C NOW CONTAINS C+.5*DTI*Y AND Y CONTAINS C-.5*DTI*Y

THIS PART WILL DO THE CHOLESKY DECOMPOSITION OF
THE MATRIX <C> WHICH IS ALREADY PUT IN THE BANDED
FORM. ORIGINAL VERSION WAS CODED BY PROF. CANTIN

```

DO 300 I = 1, NNT
DUM = C(I,1)
DUM = DSQRT(DUM)

```



```

00015320
00015330
00015340
00015350
00015360
00015370
00015380
00015390
00015400
00015410
00015420
00015430
00015440
00015450
00015460
00015470
00015480

```

```

C      DO 200 J = 1, NBAND
        C(I,J) = C(I,J) / DUM
      200 C
      DO 260 J = 2, NBAND
        L = I + J - 1
        IF (L.GT.NNT) GO TO 260
        AA = C(I,J)
        IF (AA.EQ.0.D0) GO TO 260
        DO 250 K = J, NBAND
          M = K - J + 1
          C(L,M) = C(L,M) - AA * C(I,K)
        250 C
        260 CONTINUE
      300 C
      RETURN
      END
C

```



```

2DIAG(37),BDRYI(16,5),BDRYO(16,5),VOL(36),VOLO(36),ENTE0(75),
3ENTE1(75),ENTIO(75),ENTII(75)
C C
C1 = .81157810217736320
C2 = 1.6755160819145570
C3 = .026179938779914950
C4 = 1.2042771838760882
C5 = -.2094395102393196
C6 = -.3665191429188094
C
PAI = 3.141592653589793
C C C C C
      FLOW IN THE LONGITUDINAL DIRECTION , AS IN PIPE
N = 0
IF (Q2.EQ.0.D0) GO TO 46
N = NCFOT
IF (KK.EQ.1) GO TO 40
IF (NCHECK.EQ.0) GO TO 45
C C
DO 41 I = 1, NCFOT
41 FT1(I) = FT2(I)
C
40 TIME = KK * DTI
C C
      M = KK / 2 * M
      I = KK - 1
      M = M + 1
      IF ((KK.LE.150).AND.(I.EQ.1)) ENTE0(M) = FT1(I)
      IF ((KK.LE.150).AND.(I.EQ.1)) ENTIO(M) = TIME-DTI
C C
DO 42 I = 2, NRPP0
42 IF (TIME.GT.BDRYO(I,1)) VEL = BDRYO(I,5)
C
IF (VEL.EQ.0.D0) GO TO 46
C C
IF (KK.GT.1) GO TO 75
NECFO = (NCFOT-1) / 2
C C
DUM = 0.D0
DO 10 I=2,NCFOT

```

```

00015950
00015960
00015970
00015980
00015990
00016000
00016010
00016020
00016030
00016040
00016050
00016060
00016070
00016080
00016090
00016100
00016110
00016120
00016130
00016140
00016150
00016160
00016170
00016180
00016190
00016200
00016210
00016220
00016230
00016240
00016250
00016260
00016270
00016280
00016290
00016300
00016310
00016320
00016330
00016340
00016350
00016360
00016370
00016380
00016390
00016400
00016410
00016420

```



```

R1 = R(NCFO(I-1))
R2 = R(NCFO(I))
IF (R1.GE.R2) DUM = R1
10 CONTINUE
IF (DUM.EQ.0.D0) DUM = R(NCFO(NCFOT))

C C C
DUM = DUM + 1.D0
AREAO = PAI * DUM**2
VOLO(1) = 0.D0
DO 70 I = 1, NECFO
K = 2 * I - 1
J1 = NCFO(K)
J2 = NCFO(K+1)
J3 = NCFO(K+2)

C
DUM = C1*R(J1)**2+C2*R(J2)**2+C3*R(J3)**2
DUM=DUM+C4*R(J1)*R(J2)+C5*R(J1)*R(J3)+C6*R(J2)*R(J3)
VOLO(K+1) = (AREAO-DUM)*DABS(Z(J2)-Z(J1))

C C
DUM = C3*R(J1)**2+C2*R(J2)**2+C1*R(J3)**2
DUM=DUM+C6*R(J1)*R(J2)+C5*R(J1)*R(J3)+C4*R(J2)*R(J3)
VOLO(K+2) = (AREAO-DUM)*DABS(Z(J3)-Z(J2))
70 CONTINUE

C C
DO 73 I = 2, NRPP0
IF (TIME.GT.BDRYO(I,1)) KK2 = I
73 CONTINUE
TIM = TIME
TF = BDRYO(KK2,4) * (TIM-BDRYO(KK2,1))
FT2(1) = BDRYO(KK2,2) + TF

C C C
DUMY = VEL*(AREAO-PAI*R(NCFO(1))**2)

C C
DO 74 I = 2, NCFOT
TIM1 = VOLO(I) / DUMY
TIM = TIM - TIM1
DO 76 J = 2, NRPP0
IF (TIM.GE.BDRYO(J,1)) KK2 = J
76 CONTINUE

C C
TF = BDRYO(KK2,4) * (TIM-BDRYO(KK2,1))

```



```

74 FT2(I) = BDRYO(KK2,2) + TF
   CONTINUE
   GO TO 46
45 TIME = KK * DTI
46 IF (Q3.EQ.0.D0) GO TO 56
   IF (KK.EQ.1) GO TO 50
   IF(NCHECK.EQ.0) GO TO 55
   DO 51 I = 1, NCFIT
   J = N + I
51 FT1(J) = FT2(J)
   50 TIME = KK * DTI
   C
   C
   M = KK / 2
   I = KK - 2 * M
   M = M + 1
   IF ((KK.LE.150).AND.(I.EQ.1)) ENTEI(M) = FT1(N+1)
   IF ((KK.LE.150).AND.(I.EQ.1)) ENTEII(M) = TIME-DTI
   C
   DO 52 I = 2, NRPPI
   52 IF (TIME.GT.BDRYI(I,1)) VEL = BDRYI(I,5)
   IF (VEL.EQ.0.D0) GO TO 56
   IF (KK.GT.1) GO TO 82
   NECFI = (NCFIT-1)/2
   AREA = PAI * R(NCFI(1))**2
   VOL(1) = 0.D0
   C
   DO 81 I = 1, NECFI
   K = 2 * I - 1
   J1 = NCFI(K)
   J2 = NCFI(K+1)
   J3 = NCFI(K+2)
   DUM = C1*R(J1)**2+C2*R(J2)**2+C3*R(J3)**2
   DUM=DUM+C4*R(J1)*R(J2)+C5*R(J1)*R(J3)+C6*R(J2)*R(J3)
   VOL(K+1) = DUM * DABS(Z(J2)-Z(J1))
   C
   C
   DUM = C3*R(J1)**2+C2*R(J2)**2+C1*R(J3)**2

```



```

      DUM=DUM+C6*R(J1)*R(J2)+C5*R(J1)*R(J3)+C4*R(J2)*R(J3)
      VOL(K+2) = DUM * DABS(Z(J3) - Z(J2))
81  CONTINUE
C
C
C
C
C
      DO 83 I = 2 , NRPPI
      IF (TIME.GT .BDRYI(I,1)) KK2 = I
82  CONTINUE
C
      TIM = TIME
      TF = BDRYI(KK2,4) * (TIM-BDRYI(KK2,1))
      FT2(N+1) = BDRYI(KK2,2) + TF
C
      DUMY = VEL * AREA
C
C
      DO 84 I = 2 , NCFIT
      TIM1 = VOL(I) / DUMY
      TIM = TIM - TIM1
      DO 85 J = 2 , NRPPI
      IF (TIM.GE.BDRYI(J,1)) KK2 = J
84  CONTINUE
      TF = BDRYI(KK2,4) * (TIM-BDRYI(KK2,1))
      FT2(N+1) = BDRYI(KK2,2) + TF
85  CONTINUE
      GO TO 56
C
C
      TIME = KK * DTI
85  RETURN
86  END
C
C

```

```

00017390
00017400
00017410
00017420
00017430
00017440
00017450
00017460
00017470
00017480
00017490
00017500
00017510
00017520
00017530
00017540
00017550
00017560
00017570
00017580
00017590
00017600
00017610
00017620
00017630
00017640
00017650
00017660
00017670
00017680
00017690
00017700
00017710
00017720
00017730
00017740

```


00017750

SUBROUTINE FORMV

*
* THE VECTOR V OF THE RIGHT HAND SIDE OF THE DISCRETIZED
* FINITE ELEMENT FORMULATION FOR CONVECTION OR CONSTANT
* TEMPERATURE (OUTSIDE OR INSIDE) BOUNDARY CONDITION IS
* FORMED HERE.

IMPLICIT REAL * 8 (A-H,O-Z)

COMMON /ONE/ AJ(2,2),B(4,16),COPY(37,298),D(4,4),DL(4,4),
1DNZR(2,8),ESM(16,16),F(24,298),SSM(298,66),STOR(20,150)

COMMON /TWO/ AL(5),DENS(5),DETA(8),DXI(8),E(5),EPSO(4),
1FE(16),GC(4),GX(4),POI(5),R(149),SHP(8),SHT(5),TK(5),Z(149)

COMMON /TRE/ AXIALF,DTI,EVERY,FORS1,OMEGA,HTCI,HTCO,
1POSI,PRES,Q1,Q2,Q3,SHMAX,SMAX,TIME,tIME1,tINIT,tMAXM,TREF,
2TMAXO,RMAXM,RMAXO,ZMAXM,ZMAXO,Q4,BIG

COMMON /FOR/ NN(40,9)

COMMON /FIV/ NCFI(37),NCFO(37),NNR(37),NRE(37),NNL(37),

1NPN(37)
COMMON /SIX/ IBAND,IEXT,INTP,IVEC,JVEC,KK,LOAD,NCFIT,NCFOT,
1NCHECK,NET,NDF,NGP,NNE,NNLT,NNRT,NPASS,NPROB,NRPP1,NRPP0,
3NNT,NNRE,NPNT,IPLANE,NBAND

EQUIVALENCE (C,SSM), (Y,SSM(1,18)), (T,SSM(1,35)),
1(BTE,SSM(1,36)), (TEMP,SSM(1,37)), (V,SSM(1,38)),
2(FT1,SSM(1,39)), (FT2,SSM(1,40)), (CE,SSM(1,41)),
3(YE,SSM(1,42)), (OFF1,SSM(1,43)), (OFF2,SSM(1,44)),
4(DIAG,SSM(1,45)), (BDRY1,SSM(1,46)), (BDRYO,SSM(1,47)),
5(ENT11,SSM(1,48)), (ENT10,SSM(1,49)), (ENTEI,SSM(1,50)),
6(ENTEO,SSM(1,51)), (VOL,SSM(1,52)), (VOLO,SSM(1,53))

DIMENSION C(149,33),Y(149,33),T(149),BTE(149),TEMP(149),
1V(149),FT1(90),FT2(90),CE(8,8),YE(8,8),OFF1(37),OFF2(37),
2DIAG(37),BDRY1(16,5),BDRYO(16,5),VOL(36),VOLO(36),ENTE0(75),
3ENTE1(75),ENT10(75),ENT11(75)

00017760
00017770
00017780
00017790
00017800
00017810
00017820
00017830
00017840
00017850
00017860
00017870
00017880
00017890
00017900
00017910
00017920
00017930
00017940
00017950
00017960
00017970
00017980
00017990
00018000
00018010
00018020
00018030
00018040
00018050
00018060
00018070
00018080
00018090
00018100
00018110
00018120
00018130
00018140
00018150
00018160
00018170
00018180
00018190
00018200

00018210
00018220
00018230
00018240
00018250
00018260
00018270
00018280
00018290
00018300
00018310
00018320
00018330
00018340
00018350
00018360
00018370
00018380
00018390
00018400
00018410
00018420
00018430
00018440
00018450
00018460
00018470
00018480
00018490
00018500
00018510
00018520
00018530
00018540
00018550
00018560
00018570
00018580
00018590
00018600
00018610
00018620
00018630
00018640
00018650
00018660
00018670
00018680

C

N = 0
IF (Q2.GT.0.D0) GO TO 5
GO TO 6

5 N = NCFOT
DO 1 I = 1, NCFOT
J = NCFOT(I)
1 V(J) = 0.D0

C

DO 2 J = 1, NCFOT
I = NCFOT(J)
2 V(I) = V(I) + DIAG(J) * (FT1(J)+FT2(J))

C

N1 = NCFOT - 1
DO 3 I = 1, N1
I1 = I + 1
K1 = NCFOT(I1)
K2 = NCFOT(I1+1)
V(K1) = V(K1) + OFF1(I) * (FT1(I1)+FT2(I1))
3 V(K2) = V(K2) + OFF1(I) * (FT1(I1)+FT2(I1))

C

N2 = NCFOT - 2
DO 4 I = 1, N2, 2
I2 = I + 2
K2 = NCFOT(I2)
J = NCFOT(I)
V(J) = V(J) + OFF2(I) * (FT1(I2)+FT2(I2))
4 V(K2) = V(K2) + OFF2(I) * (FT1(I2)+FT2(I2))

C

6 IF (Q3.GT.0.D0) GO TO 11
GO TO 12
11 DO 7 I = 1, NCFIT
J = NCFIT(I)
7 V(J) = 0.D0

C

DO 8 J = 1, NCFIT
I = NCFIT(J)
K = N + J
8 V(I) = V(I) + DIAG(K) * (FT1(K)+FT2(K))

C

N1 = NCFIT - 1
DO 9 I = 1, N1
K = N + I
I1 = K + 1
K1 = NCFIT(I1)
K2 = NCFIT(I1+1)

00018690
 00018700
 00018710
 00018720
 00018730
 00018740
 00018750
 00018760
 00018770
 00018780
 00018790
 00018800
 00018810
 00018820
 00018830

```

C
  V(K1) = V(K1) + OFF1(K) * (FT1(I1)+FT2(I1))
  9 V(K2) = V(K2) + OFF1(K) * (FT1(K)+FT2(K))
    N2 = NCFIT - 2
    DO 10 I = 1, N2, 2
      K = N + I
      I2 = K + 2
      J = NCFI(I)
      K2 = NCFI(I+2)
      V(J) = V(J) + OFF2(K) * (FT1(I2)+FT2(I2))
      10 V(K2) = V(K2) + OFF2(K) * (FT1(K)+FT2(K))
C
  12 CONTINUE
    RETURN
  END

```


SUBROUTINE TEMPER

 IN THIS SUBROUTINE THE NODAL TEMPERATURES ARE
 EVALUATED BY TRAPEZOIDAL INTEGRATION. ALSO FOR
 EVERY 10 STEPS OF TIME INTEGRATION THE
 CORRECTION IS APPLIED .

IMPLICIT REAL * 8 (A-H,O-Z)

COMMON /ONE/ AJ(2,2),B(4,16),COPY(37,298),D(4,4),DL(4,4),
 1DNZR(2,8),ESM(16,16),F(24,298),SSM(298,66),STOR(20,150)

COMMON /TWO/ AL(5),DENS(5),DETA(8),DXI(8),E(5),EPSO(4),
 1FE(16),GC(4),GX(4),POI(5),R(149),SHP(8),SHI(5),TK(5),Z(149)

COMMON /TREF/ AXIALF,DTI,EVERY,FORS1,OMEGA,HTCI,HTCO,
 1POSI,PPRES,Q1,Q2,Q3,SHMAX,SMAX,TIME1,TIMEI,TMAXM,TREF,
 2TMAXU,RMAXM,RMAXO,ZMAXM,ZMAXO,Q4,BIG

COMMON /FOR/ NN(40,9)

COMMON /FIV/ NCFI(37),NCFO(37),NNR(37),NRE(37),NNL(37),

1NPN(37)
 COMMON /SIX/ IBAND,IEXT,INTP,IVEC,JVEC,KK,LOAD,NCFIT,NCFOT,
 1NCHECK,NET,NDF,NGP,NNE,NNLT,NNRT,NPASS,NPROB,NRPPI,NRPPO,
 3NNNT,NNRE,NPNT,IPLANE,NBAND

EQUIVALENCE (C,SSM), (Y,SSM(1,18)), (T,SSM(1,35)),
 1(BTE,SSM(1,36)), (TEMP,SSM(1,37)), (V,SSM(1,38)),
 2(BTI,SSM(1,39)), (FT2,SSM(1,40)), (CE,SSM(1,41)),
 3(YE,SSM(1,42)), (OFF1,SSM(1,43)), (OFF2,SSM(1,44)),
 4(DIAG,SSM(1,45)), (BDRYI,SSM(1,46)), (BDRYO,SSM(1,47)),
 5(ENTII,SSM(1,48)), (ENTIO,SSM(1,49)), (ENTEI,SSM(1,50)),
 6(ENTEO,SSM(1,51)), (VOL,SSM(1,52)), (VOLO,SSM(1,53))

DIMENSION C(149,33),Y(149,33),T(149),BTE(149),TEMP(149),
 1V(149),FT1(90),FT2(90),CE(8,8),YE(8,8),OFF1(37),OFF2(37),
 2DIAG(37),BDRYI(16,5),BDRYO(16,5),VOL(36),VOLO(36),ENTEO(75),
 3ENTEI(75),ENTIO(75),ENTII(75)

00018840
 00018850
 00018860
 00018870
 00018880
 00018890
 00018900
 00018910
 00018920
 00018930
 00018940
 00018950
 00018960
 00018970
 00018980
 00018990
 00019000
 00019010
 00019020
 00019030
 00019040
 00019050
 00019060
 00019070
 00019080
 00019090
 00019100
 00019110
 00019120
 00019130
 00019140
 00019150
 00019160
 00019170
 00019180
 00019190
 00019200
 00019210
 00019220
 00019230
 00019240
 00019250
 00019260
 00019270
 00019280
 00019290


```

00019300
00019310
00019320
00019330
00019340
00019350
00019360
00019370
00019380
00019390
00019400
00019410
00019420
00019430
00019440
00019450
00019460
00019470
00019480
00019490
00019500
00019510
00019520
00019530
00019540
00019550
00019560
00019570
00019580
00019590
00019600
00019610
00019620
00019630
00019640
00019650
00019660
00019670
00019680
00019690
00019700
00019710
00019720
00019730
00019740
00019750
00019760
00019770

```

```

NCHECK = 1

```

```

DUM = DT1 / 2. DO
DO 1 I=1,NNT
1 V(I) = DUM * V(I)

```

THIS PART WILL MULTIPLY THE BANDED MATRIX Y BY THE
COLUMN VECTOR OF TEMPERATURE AND ADD TO V.

```

DO 7 J = 1, NNT
BTE(J) = 0. DO
M = MINO(NBAND, NNT+1-J)
DO 5 K = 1, M
L = J + K - 1
BTE(J) = BTE(J) + Y(J,K) * TEMP(L)
5 CONTINUE
IF (J .LT. 2) GO TO 7
L = MINO(J, NBAND)
DO 6 K = 2, L
M = J - K + 1
BTE(J) = BTE(J) + Y(M,K) * TEMP(M)
6 CONTINUE
7 BTE(J) = BTE(J) + V(J)

```

SOLVING FOR TEMPERATURE BY FORWARD AND BACK SUBSTITUTION

```

DO 32 I = 1, NNT
BTE(I) = BTE(I) / C(I,1)
DO 31 J = 2, NBAND
L = I + J - 1
IF (L .GT. NNT) GO TO 32
DUM = C(I,J)
31 BTE(L) = BTE(L) - DUM * BTE(I)
32 CONTINUE
BTE(NNT) = BTE(NNT) / C(NNT,1)
DO 34 L = 2, NNT
K = NNT - L + 1
SUM = 0. DO
DO 33 J = 2, NBAND

```


00019780
00019790
00019800
00019810
00019820
00019830
00019840
00019850
00019860
00019870
00019880
00019890
00019900
00019910
00019920
00019930
00019940
00019950
00019960
00019970
00019980
00019990
00020000
00020010
00020020
00020030
00020040
00020050
00020060
00020070
00020080
00020090
00020100
00020110
00020120
00020130
00020140
00020150
00020160
00020170
00020180
00020190
00020200
00020210
00020220
00020230
00020240
00020250

```

M = J + K - 1
IF (NNT .LT. M) GO TO 34
33 SUM = SUM + C(K,J) * BTE(M)
34 BTE(K) = (BTE(K) - SUM) / C(K,1)
C

IF (KK.GT.2) GO TO 2
K0 = 0
C
C
APPLICATION OF IRONS CORRECTION.
C
2 K1 = K0 + 12 - KK
GO TO (70, 60, 50), K1
GO TO 14
C
50 DO 51 I = 1, NNT
51 T(I) = .25D0 * BTE(I)
GO TO 14
C
60 DO 61 I = 1, NNT
61 T(I) = T(I) + .5D0 * BTE(I)
GO TO 14
C
70 DO 71 I = 1, NNT
71 TEMP(I) = T(I) + .25D0 * BTE(I)
C
TEMPERATURE IS CORRECTED NOW
C
NCHECK = 0
K0 = K0 + 10
KK = KK - 1
TIME = KK * DTI
IP = 0
C
GO TO 16
14 DO 4 I = 1, NNT
4 TEMP(I) = BTE(I)
IP = 1
C
16 CONTINUE
C
C1 = 1000.D0 * TIME
DO 21 J = 1, IVEC
21 C2 = (TIME1 + EVERY * (J-1)) * 1000.D0
DC = DABS(C1 - C2)
IF (DC.LE..0001) GO TO 20
21 CONTINUE
C

```


00020260
00020270
00020280
00020290
00020300
00020310
00020320
00020330
00020340
00020350
00020360
00020370
00020380
00020390
00020400
00020410
00020420
00020430
00020440
00020450
00020460
00020470
00020480
00020490
00020500
00020510
00020520
00020530
00020540
00020550
00020560
00020570

```

C      GO TO 22
C
C      DO 40 I = 1, NNT
C      40 STOR(J,I) = TEMP(I)
C      STOR(J,NNT+1) = TIME
C
C      22 IF (INIP.EQ.0) GO TO 81
C      INT1 = INIP
C      83 KUNT = KK / INT1
C      IF (KUNT.EQ.0) GO TO 81
C      KUNT = KK - KUNT * INT1
C      IF (KUNT.EQ.0) GO TO 82
C      INT1 = INT1 + INIP
C      GO TO 83
C
C      82 ID = INIP / 10 * 10
C      IF ((ID.EQ.0).AND.(IP.NE.0)) GO TO 81
C      WRITE (6,101) TIME
C      101 FORMAT (/,5X,'AT TIME=',F8.2,' THE TEMPERATURES AT THE ',
C      222 1,NODES ARE: ',/ ) (I,TEMP(I),I=1,NNT)
C      81 WRITE (6,222) (I,13,' ',1PD13.5))
C      81 KK = KK + 1
C
C      RETURN
C      END
C

```



```

SUBROUTINE STIFF
** ** ** ** **
SUBROUTINE STIFF, CALCULATES THE STIFFNESS MATRIX
FOR A GIVEN PROBLEM IN THE BANDED FORM, APPLIES THE
REQUIRED BOUNDARY CONDITIONS AND AT THE END MAKES
THE CHOLESKY DECOMPOSITION.
** ** ** **

IMPLICIT REAL * 8 (A-H,O-Z)

COMMON /ONE/ AJ(2,2), B(4,16), COPY(37,298), D(4,4), DL(4,4),
1DNZR(2,8), ESM(16,16), F(24,298), SSM(298,66), STOR(20,150)

COMMON /TWO/ AL(5), DENS(5), DELTA(8), DXI(8), E(5), EPSO(4),
1FE(16), GC(4), GX(4), POI(5), R(149), SHP(8), SHT(5), TK(5), Z(149)

COMMON /TRE/ AXIALF, DTI, EVERY, FORS1, OMEGA, HTCI, HTCO,
1POSTI, PRES, Q1, Q2, Q3, SHMAX, SMAX, TIMEI, TINIT, TMAXM, TREF,
2TMAXO, RMAXM, RMAXO, ZMAXM, ZMAXO, Q4, BIG

COMMON /FOR/ NN(40,9)

COMMON /FIV/ NCFI(37), NCFO(37), NNR(37), NRE(37),>NNL(37),
1NPN(37)

COMMON /SIX/ IBAND, IEXT, INTP, IVEC, JVEC, KK, LOAD, NCFIT, NCFOT,
1INCHECK, NET, NDF, NGP, NNE,>NNLT, NNRT, NPASS, NPROB, NRPPI, NRPPPO,
3NNNT, NNRE, NPNT, IPLANE, NBAND

DO 5 I = 1, NDF
DO 5 J = 1, IBAND
5 SSM(I,J) = 0.0
PAI = 3.141592653589793
DO 31 L = 1, NET
IF (L.EQ.1) GO TO 1
IF (NN(L,9).EQ>NN(L-1,9)) GO TO 3
1 K = NN(L,9)
EL = 1.00 - POI(K)
FACT = (E(K)*EL)/((1.00+POI(K))*(1.00-2.00*POI(K)))

```



```

C      EL = POI(K) / EL
C      D(1,1) = FACT
C      D(1,2) = EL*FACT
C      D(1,3) = D(1,2)
C      D(1,4) = 0.D0
C      D(2,2) = FACT
C      D(2,3) = D(1,2)
C      D(2,4) = 0.D0
C      D(3,3) = FACT
C      D(3,4) = 0.D0
C      D(4,4) = E(K)/(2.D0*(1.D0+POI(K)))
C
C      DO 27 I = 1, 4
C      DO 27 J = I, 4
C      27 D(J,I) = D(I,J)
C
C      CALL LLT (4,D,DL)
C
C      DL IS THE LOWER TRIANGLE OF DECOMPOSED D.
C
C      3 CONTINUE
C      NNE2 = 2 * NNE
C      DO 11 I = 1, NNE2
C      DO 11 J = 1, NNE2
C      11 ESM(I,J) = 0.D0
C
C      DO 9 N = 1,NGP
C      ETA = GX(N)
C      DO 8 M = 1,NGP
C      XI = GX(M)
C
C      DO 7 I = 1,4
C      DO 7 J = 1,NNE2
C      7 B(I,J) = 0.D0
C
C      SHP(1) = .25D0 * (1.D0-XI) * (1.D0-ETA) * (-XI-ETA-1.D0)
C      SHP(2) = .5D0 * (1.D0-XI**2) * (1.D0-ETA)
C      SHP(3) = .25D0 * (1.D0+XI) * (1.D0-ETA) * (XI-ETA-1.D0)

```



```

SHP(4) = .5D0 * (1.D0-ETA**2) * (1.D0+XI)      (XI+ETA-1.D0)
SHP(5) = .25D0 * (1.D0+XI) * (1.D0+ETA) *
SHP(6) = .5D0 * (1.D0-XI**2) * (1.D0+ETA)
SHP(7) = .25D0 * (1.D0-XI) * (1.D0+ETA) * (-XI+ETA-1.D0)
SHP(8) = .5D0 * (1.D0-ETA**2) * (1.D0-XI)

```

CCCCC

DERIVATIVES OF THE SHAPE FUNCTIONS WITH RESPECT TO XI.

```

DXI(1) = .25D0 * (1.D0-ETA) * (2.D0*XI+ETA)
DXI(2) = -.XI * (1.D0-ETA)
DXI(3) = .25D0 * (1.D0-ETA) * (2.D0*XI-ETA)
DXI(4) = .5D0 * (1.D0-ETA**2)
DXI(5) = .25D0 * (1.D0+ETA) * (2.D0*XI+ETA)
DXI(6) = -.XI * (1.D0+ETA)
DXI(7) = .25D0 * (1.D0+ETA) * (2.D0*XI-ETA)
DXI(8) = -.5D0 * (1.D0-ETA**2)

```

CCCCC

DERIVATIVES OF THE SHAPE FUNCTIONS WITH RESPECT TO ETA.

```

DETA(1) = .25D0 * (1.D0-XI) * (2.D0*ETA+XI)
DETA(2) = -.5D0 * (1.D0-XI**2) * (2.D0*ETA+XI)
DETA(3) = .25D0 * (1.D0+XI) * (2.D0*ETA-XI)
DETA(4) = -.ETA * (1.D0+XI)
DETA(5) = .25D0 * (1.D0+XI) * (2.D0*ETA+XI)
DETA(6) = .5D0 * (1.D0-XI**2)
DETA(7) = .25D0 * (1.D0-XI) * (2.D0*ETA-XI)
DETA(8) = -.ETA * (1.D0-XI)

```

CCCCC

CONSTRUCTION OF THE JACOBIAN MATRIX.

```

DO 20 I = 1, 2
DO 20 J = 1, 2
20 AJ(I,J) = 0.D0

```

CC

```

DO 22 I = 1, NNE
AJ(1,1) = AJ(1,1) + DXI(1) * Z(NN(L,I))
AJ(1,2) = AJ(1,2) + DXI(1) * R(NN(L,I))
AJ(2,1) = AJ(2,1) + DETA(1) * Z(NN(L,I))
AJ(2,2) = AJ(2,2) + DETA(1) * R(NN(L,I))
22 CONTINUE

```

C

```

00021520
00021530
00021540
00021550
00021560
00021570
00021580
00021590
00021600
00021610
00021620
00021630
00021640
00021650
00021660
00021670
00021680
00021690
00021700
00021710
00021720
00021730
00021740
00021750
00021760
00021770
00021780
00021790
00021800
00021810
00021820
00021830
00021840
00021850
00021860
00021870
00021880
00021890
00021900
00021910
00021920
00021930
00021940
00021950
00021960
00021970
00021980
00021990

```



```

C      DET =AJ(1,1) * AJ(2,2) - AJ(1,2) * AJ(2,1)
C      DEI =1.0D0/DET
C
C      INVERTING THE JACOBIAN AND STORING IN AJ AGAIN.
C
C      DUM = AJ(1,1) * DEI
C      AJ(1,1) = AJ(2,2) * DEI
C      AJ(1,2) = -AJ(1,2) * DEI
C      AJ(2,1) = -AJ(2,1) * DEI
C      AJ(2,2) = DUM
C
C      CR = 0.D0
C      DO 2 I = 1,NNE
C      M1 = NN(L,I)
C      2 CR = CR + SHP(I) * R(M1)
C
C      EVALUATION OF THE B MATRIX
C
C      DO 15 I = 1, NNE
C      K1 = 2 * I - 1
C      K2 = K1 + 1
C      B(1,K2) = AJ(1,1) * DXI(I) + AJ(1,2) * DETA(I)
C      B(2,K1) = AJ(2,1) * DXI(I) + AJ(2,2) * DETA(I)
C      B(3,K1) = SHP(I) / CR
C      B(4,K1) = B(1,K2)
C      B(4,K2) = B(2,K1)
C      15 CONTINUE
C
C      CALCULATION OF (DL TRANSPOSE)*B AND STORING IN <B>
C
C      DO 16 I = 1,4
C      DO 13 J = 1,NNE2
C      DUM = 0.D0
C      DO 17 K = I,4
C      17 DUM = DUM + DL(K,I) * B(K,J)
C      13 B(I,J) = DUM
C      16 CONTINUE
C
00022000
00022010
00022020
00022030
00022040
00022050
00022060
00022070
00022080
00022090
00022100
00022110
00022120
00022130
00022140
00022150
00022160
00022170
00022180
00022190
00022200
00022210
00022220
00022230
00022240
00022250
00022260
00022270
00022280
00022290
00022300
00022310
00022320
00022330
00022340
00022350
00022360
00022370
00022380
00022390
00022400
00022410
00022420
00022430
00022440
00022450
00022460
00022470

```


NOTE : NOW CONTAINS THE PRODUCT OF (DL TRANSPOSE)*B
 CALCULATION OF <BT>*<D>* AND ELEMENT STIFFNESS MATRIX
 <E S M>

DUM1 = 2*DO * PAI * CR * DET * GC(N) * GC(M)

DO 19 I = 1, NNE2
 DO 19 J = 1, I

G = 0.0D0

DO 18 K = 1, 4
 G = G + B(K,I) * B(K,J)

ESM(I,J) = ESM(I,J) + G * DUM1
 19 CONTINUE

DO 24 I = 1, NNE2

DO 24 J = 1, I
 ESM(J,I) = ESM(I,J)

8 CONTINUE
 9 CONTINUE

CONSTRUCTION OF THE SYSTEM STIFFNESS MATRIX
 IN BANDED FORM.

DO 25 I = 1, NNE
 II = 2 * (NN(L,I) - 1)

DO 25 J = 1, NNE
 JJ = 2 * (NN(L,J) - 1)
 IF (II.GT.JJ) GO TO 25
 JJ = JJ - II

DO 25 K = 1, 2

JJ = II + K
 LL = 2 * (I-1) + K

IF (J1.EQ.0) IJ = K

DO 25 JI = IJ, 2
 MI = 2 * (J-1) + JI

KK = JI + JI - K + 1
 SSM(JJ,KK) = SSM(JJ,KK) + ESM(LL,MI)

25 CONTINUE

00022480
 00022490
 00022500
 00022510
 00022520
 00022530
 00022540
 00022550
 00022560
 00022570
 00022580
 00022590
 00022600
 00022610
 00022620
 00022630
 00022640
 00022650
 00022660
 00022670
 00022680
 00022690
 00022700
 00022710
 00022720
 00022730
 00022740
 00022750
 00022760
 00022770
 00022780
 00022790
 00022800
 00022810
 00022820
 00022830
 00022840
 00022850
 00022860
 00022870
 00022880
 00022890
 00022900
 00022910
 00022920
 00022930
 00022940
 00022950


```

C 31 CONTINUE
C IF (IPLANE.EQ.0) GO TO 49
C
C PART OF SSM IS BEING STORED IN COPY.
C
C DO 41 J = 1, NDF
C DO 41 I = 1, NNRE
C 41 COPY(I,J) = 0.DO
C
C DO 44 II = 1, NNRE
C K = 2 * NRE(II)
C JJ = K
C DO 42 J = 1, IBAND
C IF (JJ.GT.NDF) GO TO 45
C COPY(II,JJ) = SSM(K,J)
C 42 JJ = JJ + 1
C
C 45 LL = K - 1
C IF (LL.EQ.0) GO TO 44
C JJ = LL
C
C DO 43 J = 1, LL GO TO 44
C IF (J.EQ.IBAND) GO TO 44
C COPY(II,JJ) = SSM(JJ,J+1)
C 43 JJ = JJ - 1
C 44 CONTINUE
C
C 49 CONTINUE
C
C IF (NNRT.EQ.0) GO TO 33
C DO 34 I = 1, NNRT
C K = 2 * NNR(I) - 1
C 34 SSM(K,1) = SSM(K,1) * BIG
C
C 33 IF (NNLT.EQ.0) GO TO 35
C DO 36 I = 1, NNLT
C K = 2 * NNL(I)
C 36 SSM(K,1) = SSM(K,1) * BIG
C 35 CONTINUE

```

```

00022960
00022970
00022980
00022990
00023000
00023010
00023020
00023030
00023040
00023050
00023060
00023070
00023080
00023090
00023100
00023110
00023120
00023130
00023140
00023150
00023160
00023170
00023180
00023190
00023200
00023210
00023220
00023230
00023240
00023250
00023260
00023270
00023280
00023290
00023300
00023310
00023320
00023330
00023340
00023350
00023360
00023370
00023380
00023390
00023400
00023410
00023420
00023430

```


DECOMPOSING SYSTEM STIFFNESS MATRIX

```

DO 300 I = 1, NDF
DUM = SSM(I,1)
DUM = DSQRT(DUM)
DO 200 J = 1, IBAND
SSM(I,J) = SSM(I,J) / DUM

DO 260 J = 2, IBAND
L = I + J - 1
IF (L.GT.NDF) GO TO 260
AA = SSM(I,J)
IF (AA.EQ.0.D0) GO TO 260
DO 250 K = J, IBAND
M = K - J + 1
SSM(L,M) = SSM(L,M) - AA * SSM(I,K)
250 CONTINUE
260 CONTINUE
300 RETURN
350 END

```

00023440
00023450
00023460
00023470
00023480
00023490
00023500
00023510
00023520
00023530
00023540
00023550
00023560
00023570
00023580
00023590
00023600
00023610
00023620
00023630
00023640
00023650
00023660

SUBROUTINE FORMF

 THE THERMAL LOAD VECTORS ARE BEING CALCULATED HERE
 AND FOR THE BOUNDARY CONDITION "ENDS REMAIN PLANE"
 ANOTHER LOAD VECTOR FOR THE UNIT END DISPLACEMENT
 IS BEING CALCULATED.

IMPLICIT REAL * 8 (A-H,O-Z)

COMMON /ONE/ AJ(2,2),B(4,16),COPY(37,298),D(4,4),DL(4,4),
 1DNZR(2,8),ESM(16,16),F(24,298),SSM(298,66),STOR(20,150)
 COMMON /TWO/ AL(5),DENS(5),DETA(8),DXI(8),E(5),EPSO(4),
 1FE(16),GC(4),GX(4),POI(5),R(149),SHP(8),SHT(5),TK(5),Z(149)
 COMMON /TRE/ AXIALF,DTI,EVERY,FORS1,OMEGA,HTCI,HTCO,
 1POSI,PRES,Q1,Q2,Q3,SHMAX,SMAX,TIME,TIME1,INIT,TMAXM,TREF,
 2TMAXO,RMAXM,RMAXO,ZMAXM,ZMAXO,Q4,BIG

COMMON /FOR/ NN(40,9)

COMMON /FIV/ NCFI(37),NCFO(37),NNR(37),NRE(37),NNL(37),
 1NPN(37)
 COMMON /SIX/ IBAND,IEXT,INTP,IVEC,JVEC,KK,LOAD,NCFIT,NCFOT,
 1NCHECK,NET,NDF,NGP,NNE,NNLT,NNRT,NPASS,NPROB,NRPPI,NRPPO,
 3NNNT,NNRE,NPNT,IPLANE,NBAND

PAI = 3.141592653589793

IDUM = IVEC + JVEC
 IF (LOAD.NE.0) IDUM = IDUM - 1
 DO 1 I = 1, IDUM
 DO 1 J = 1, NDF
 1 F(I,J) = 0.00

IDUM = IDUM + 1
 IF (LOAD.NE.0) IDUM = IDUM + 1
 DO 4 I = 1, NDF
 4 F(IDUM,I) = 0.00

NNE2 = 2 * NNE


```

C      DO 25 L = 1 , NET
C
C      DO 9 N = 1,NGP
C      ETA = GX(N)
C      DO 8 M = 1,NGP
C      XI = GX(M)
C
C      DO 7 I = 1,4
C      DO 7 J = 1,NNE2
C      B(I,J) = 0.D0
C
C      SHP(1) = .25D0 * (1.D0-XI) * (1.D0-ETA) * (-XI-ETA-1.D0)
C      SHP(2) = .5D0 * (1.D0-XI**2) * (1.D0-ETA)
C      SHP(3) = .25D0 * (1.D0+XI) * (1.D0-ETA) * (XI-ETA-1.D0)
C      SHP(4) = .5D0 * (1.D0-ETA**2) * (1.D0+XI)
C      SHP(5) = .25D0 * (1.D0+XI) * (1.D0+ETA) * (XI+ETA-1.D0)
C      SHP(6) = .5D0 * (1.D0-XI**2) * (1.D0+ETA)
C      SHP(7) = .25D0 * (1.D0-XI) * (1.D0+ETA) * (-XI+ETA-1.D0)
C      SHP(8) = .5D0 * (1.D0-ETA**2) * (1.D0-XI)
C
C      DERIVATIVES OF THE SHAPE FUNCTIONS WITH RESPECT TO XI.
C
C      DXI(1) = .25D0 * (1.D0-ETA) * (2.D0*XI+ETA)
C      DXI(2) = -.XI * (1.D0-ETA)
C      DXI(3) = .25D0 * (1.D0-ETA) * (2.D0*XI-ETA)
C      DXI(4) = .5D0 * (1.D0-ETA**2)
C      DXI(5) = .25D0 * (1.D0+ETA) * (2.D0*XI+ETA)
C      DXI(6) = -.XI * (1.D0+ETA)
C      DXI(7) = .25D0 * (1.D0+ETA) * (2.D0*XI-ETA)
C      DXI(8) = -.5D0 * (1.D0-ETA**2)
C
C      DERIVATIVES OF THE SHAPE FUNCTIONS WITH RESPECT TO ETA.
C
C      DETA(1) = .25D0 * (1.D0-XI) * (2.D0*ETA+XI)
C      DETA(2) = -.5D0 * (1.D0-XI**2)
C      DETA(3) = .25D0 * (1.D0+XI) * (2.D0*ETA-XI)
C      DETA(4) = -.ETA * (1.D0+XI)
C      DETA(5) = .25D0 * (1.D0+XI) * (2.D0*ETA+XI)

```



```

00024610
00024620
00024630
00024640
00024650
00024660
00024670
00024680
00024690
00024700
00024710
00024720
00024730
00024740
00024750
00024760
00024770
00024780
00024790
00024800
00024810
00024820
00024830
00024840
00024850
00024860
00024870
00024880
00024890
00024900
00024910
00024920
00024930
00024940
00024950
00024960
00024970
00024980
00024990
00025000
00025010
00025020
00025030
00025040
00025050
00025060
00025070
00025080

DETA(6) = .5D0 * (1.D0-XI**2)
DETA(7) = .25D0 * (1.D0-XI) * (2.D0*ETA-XI)
DETA(8) = -ETA * (1.D0-XI)

      CONSTRUCTION OF THE JACOBIAN MATRIX.

      DO 20 I = 1, 2
      DO 20 J = 1, 2
      20 AJ(I,J) = 0.D0

      DO 22 I = 1, NNE
      AJ(1,1) = AJ(1,1) + DXI(I) * Z(NN(L,I))
      AJ(1,2) = AJ(1,2) + DXI(I) * R(NN(L,I))
      AJ(2,1) = AJ(2,1) + DETA(I) * Z(NN(L,I))
      AJ(2,2) = AJ(2,2) + DETA(I) * R(NN(L,I))
      22 CONTINUE

      DET = AJ(1,1) * AJ(2,2) - AJ(1,2) * AJ(2,1)
      DEI = 1.0D0/DET

      INVERTING THE JACOBIAN AND STORING IN AJ AGAIN.

      DUM = AJ(1,1) * DEI
      AJ(1,1) = AJ(2,2) * DEI
      AJ(1,2) = -AJ(1,2) * DEI
      AJ(2,1) = -AJ(2,1) * DEI
      AJ(2,2) = DUM

      CR = 0.D0
      DO 2 I = 1, NNE
      MI = NN(L,I)
      2 CR = CR + SHP(I) * R(MI)

      EVALUATION OF THE B MATRIX

      DO 15 I = 1, NNE
      K1 = 2 * I - 1
      K2 = K1 + 1
      B(1,K2) = AJ(1,1) * DXI(I) + AJ(1,2) * DETA(I)

```



```

B(2,K1) = AJ(2,1) * DXI(I) + AJ(2,2) * DETA(I)
B(3,K1) = SHP(I) / CR
B(4,K1) = B(1,K2)
B(4,K2) = B(2,K1)
15 CONTINUE
CC
DUM1 = 2.00 * PAI * CR * DET * GC(N) * GC(M)
CC
DO 30 KK = 1, IVEC
CCCC
    EVALUATING TEMPERATURE AT GAUSS POINT
    TGP = 0.00
    DO 3 I = 1, NNE
        M1 = NN(L, I)
        3 TGP = TGP + SHP(I) * (STOR(KK,M1)-TREF)
    TGP = AL(NN(L,9)) * TGP
    C
    CALCULATION OF ELEMENT INITIAL STRAIN VECTOR <EPS0>
    DO 6 I = 1, 3
        6 EPS0(I) = TGP
    C
    NOTE EPS0(4) = 0.00
    PRODUCT OF <BTRANSPQSE> * <D> * <EPS0>
    CALCULATION OF ELEMENT LOAD VECTOR ,
    DUM = (TGP*E(NN(L,9)))/(1.00-2.00*POI(NN(L,9)))
    DUM = DUM * DUM1
    DO 17 I = 1, NNE2, 2
        J = I + 1
        FE(I) = (B(2,I) + B(3,I)) * DUM
        17 FE(J) = B(1,J) * DUM
    C
    CONSTRUCTION OF SYSTEM LOAD VECTORS F
    DO 18 I = 1, NNE
        CCCCCC

```



```

00025570
00025580
00025590
00025600
00025610
00025620
00025630
00025640
00025650
00025660
00025670
00025680
00025690
00025700
00025710
00025720
00025730
00025740
00025750
00025760
00025770
00025780
00025790
00025800
00025810
00025820
00025830
00025840

```

```

C C
      I2 = 2 * I - 1
      I1 = NN(L,I) * 2 - 1
      F(KK,I1) = F(KK,I1) + FE(I2)
      F(KK,I1+1) = F(KK,I1+1) + FE(I2+1)
C C
      18 CONTINUE
      30 CONTINUE
      8 CONTINUE
      9 CONTINUE
      25 CONTINUE
C C
      IF (IPLANE.EQ.0) GO TO 40
C C
      LOAD VECTOR FOR UNIT END DISPLACEMENT
C C
      K = IDUM
      DO 41 I = 1, NNRE
      L = 2 * NRE(I)
      41 F(K,L) = COPY(I,L) * BIG
C C
      40 CONTINUE
      RETURN
      END
C C

```



```

C      DUM1 = 2.DO * PAI * DENS(K) * OMEGA**2
C      IF (Q1.EQ.BRIT) DUM1 = DUM1 / 386.0885827
C      IF (Q1.NE.BRIT) DUM1 = DUM1 / 980.665
C
C      DO 9 N = 1,NGP
C      ETA = GX(N)
C      DO 8 M = 1,NGP
C      XI = GX(M)
C
C      SHP(1) = .25D0 * (1.D0-XI) * (1.D0-ETA) * (-XI-ETA-1.D0)
C      SHP(2) = .5D0 * (1.D0-XI**2) * (1.D0-ETA) * (XI-ETA-1.D0)
C      SHP(3) = .25D0 * (1.D0+XI) * (1.D0-ETA) * (XI-ETA-1.D0)
C      SHP(4) = .5D0 * (1.D0-ETA**2) * (1.D0+XI) * (XI+ETA-1.D0)
C      SHP(5) = .25D0 * (1.D0+XI) * (1.D0+ETA) * (XI+ETA-1.D0)
C      SHP(6) = .5D0 * (1.D0-XI**2) * (1.D0+ETA) * (-XI+ETA-1.D0)
C      SHP(7) = .25D0 * (1.D0-XI) * (1.D0+ETA) * (-XI+ETA-1.D0)
C      SHP(8) = .5D0 * (1.D0-ETA**2) * (1.D0-XI)
C
C      DERIVATIVES OF THE SHAPE FUNCTIONS WITH RESPECT TO XI.
C
C      DXI(1) = .25D0 * (1.D0-ETA) * (2.D0*XI+ETA)
C      DXI(2) = -XI * (1.D0-ETA)
C      DXI(3) = .25D0 * (1.D0-ETA) * (2.D0*XI-ETA)
C      DXI(4) = .5D0 * (1.D0-ETA**2)
C      DXI(5) = .25D0 * (1.D0+ETA) * (2.D0*XI+ETA)
C      DXI(6) = -XI * (1.D0+ETA)
C      DXI(7) = .25D0 * (1.D0+ETA) * (2.D0*XI-ETA)
C      DXI(8) = -.5D0 * (1.D0-ETA**2)
C
C      DERIVATIVES OF THE SHAPE FUNCTIONS WITH RESPECT TO ETA.
C
C      DETA(1) = .25D0 * (1.D0-XI) * (2.D0*ETA+XI)
C      DETA(2) = -.5D0 * (1.D0-XI**2)
C      DETA(3) = .25D0 * (1.D0+XI) * (2.D0*ETA-XI)
C      DETA(4) = -ETA * (1.D0+XI)
C      DETA(5) = .25D0 * (1.D0+XI) * (2.D0*ETA+XI)
C      DETA(6) = .5D0 * (1.D0-XI**2)
C      DETA(7) = .25D0 * (1.D0-XI) * (2.D0*ETA-XI)
C      DETA(8) = -ETA * (1.D0-XI)
C
C      CONSTRUCTION OF THE JACOBIAN MATRIX.

```



```

C      DO 20 I = 1, 2
C      DO 20 J = 1, 2
20    AJ(I,J) = 0.D0
C
C      DO 22 I = 1, NNE
C      AJ(1,1) = AJ(1,1) + Z(NN(L,I))
C      AJ(1,2) = AJ(1,2) + DXI(I) * R(NN(L,I))
C      AJ(2,1) = AJ(2,1) + DETA(I) * R(NN(L,I))
C      AJ(2,2) = AJ(2,2) + DETA(I) * R(NN(L,I))
22    CONTINUE
C
C      DET = AJ(1,1) * AJ(2,2) - AJ(1,2) * AJ(2,1)
C      DEI = 1.0D0/DET
C
C      INVERTING THE JACOBIAN AND STORING IN AJ AGAIN.
C
C      DUM = AJ(1,1) * DEI
C      AJ(1,1) = AJ(2,2) * DEI
C      AJ(1,2) = -AJ(1,2) * DEI
C      AJ(2,1) = -AJ(2,1) * DEI
C      AJ(2,2) = DUM
C
C      CR = 0.D0
C      DO 2 I = 1, NNE
C      M1 = NN(L,I)
2    CR = CR + SHP(I) * R(M1)
C      CR = CR**2
C
C      DUM = DUM1 * CR * DET * GC(N) * GC(M)
C
C      DO 17 I = 1, NNE
C      K = 2 * I - 1
C      FE(K) = FE(K) + SHP(I) * DUM
17    CONTINUE
C
C      8 CONTINUE
C      9 CONTINUE
C

```



```

C
C      CONSTRUCTION OF SYSTEM CENTRIFUGAL LOAD VECTOR
C
      DO 18 I = 1, NNE
      I2 = 2 * I - 1
      I1 = NN(L,I) * 2 - 1
      F(I1,I1) = F(I1,I1) + FE(I2)
      18 CONTINUE
C
      25 CONTINUE
C
      RETURN
      END
C

```

```

00027270
00027280
00027290
00027300
00027310
00027320
00027330
00027340
00027350
00027360
00027370
00027380
00027390
00027400
00027410
00027420
00027430

```


SUBROUTINE PRESS

SUBROUTINE PRESS EVALUATES THE PRESSURE LOAD VECTOR
FOR THE SYSTEM WHEN THERE IS ANY PRESSURE (NE.O.DO)
ACTING ON THE BODY

IMPLICIT REAL * 8 (A-H,O-Z)

COMMON /ONE/ AJ(2,2),B(4,16),COPY(37,298),D(4,4),DL(4,4),
1DNZR(2,8),ESM(16,16),F(24,298),SSM(298,66),STOR(20,150)

COMMON /TWO/ AL(5),DENS(5),DETA(8),DXI(8),E(5),EPSO(4),
1FE(16),GC(4),GX(4),POI(5),R(149),SHP(8),SHT(5),TK(5),Z(149)

COMMON /TRE/ AXIALF,DTI,EVERY,FORS1,OMEGA,HTCI,HTCO,
1POSI,PRES,Q1,Q2,Q3,SHMAX,SMAX,TIME,TIME1,TINIT,TMAXM,TREF,
2TMAXO,RMAXM,RMAXO,ZMAXM,ZMAXO,Q4,BIG

COMMON /FOR/ NN(40,9)

COMMON /FIV/ NCFI(37),NCFO(37),NNR(37),NRE(37),NNL(37),
1NPN(37)
COMMON /SIX/ IBAND,IEXT,INTP,IVEC,JVEC,KK,LOAD,NCFIT,NCFOT,
1NCHECK,NET,NDF,NGP,NNE,NNLT,NNRT,NPASS,NPROB,NRPP1,NRPP0,
3NNT,NNRE,NPNT,IPLANE,NBAND

PAI = 3.141592653589793

L = 1
IF (OMEGA.NE.O.DO) L = L + 1
II = IVEC + L

NEUP = (NPNT-1) / 2

NEUP IS THE NUMBER OF ELEMENTS UNDER PRESSURE

00027440
00027450
00027460
00027470
00027480
00027490
00027500
00027510
00027520
00027530
00027540
00027550
00027560
00027570
00027580
00027590
00027600
00027610
00027620
00027630
00027640
00027650
00027660
00027670
00027680
00027690
00027700
00027710
00027720
00027730
00027740
00027750
00027760
00027770
00027780
00027790
00027800
00027810
00027820
00027830
00027840
00027850
00027860
00027870
00027880
00027890

00028380
00028390
00028400
00028410
00028420
00028430
00028440
00028450
00028460
00028470
00028480
00028490

```

DO 5 I=1,3
J1=2*I-1
J2=2*I
J=2*I
NPN(K+I-1) - 1
F(I1,J) = F(I1,J) + FE(J1)*FACT
F(I1,J+1) = F(I1,J+1) + FE(J2)*FACT
5 CONTINUE
10 CONTINUE
C
C
RETURN
END

```


SUBROUTINE DISPL

```

*****
THIS SUBROUTINE SOLVES FOR DISPLACEMENTS, ONCE GIVEN
THE DECOMPOSED SYSTEM STIFFNESS MATRIX AND THE LOAD
VECTORS. THE LOAD VECTORS ARE REPLACED BY THE
SOLUTIONS 'THE DISPLACEMENTS'.
*****

```

```

IMPLICIT REAL * 8 (A-H,O-Z)

```

```

COMMON /ONE/ AJ(2,2),B(4,16),COPY(37,298),D(4,4),DL(4,4),
1DNZR(2,8),ESM(16,16),F(24,298),SSM(298,66),STOR(20,150)
COMMON /TWO/ AL(5),DENS(5),DETA(8),DXI(8),E(5),EPSO(4),
1FE(16),GC(4),GX(4),POI(5),R(149),SHP(8),SHT(5),TK(5),Z(149)
COMMON /TRE/ AXIALF,DTI,EVERY,FORS1,OMEGA,HTCI,HTCO,
1POST,PRES,Q1,Q2,Q3,SHMAX,SMAX,TIME,TIME1,TINIT,TMAXM,TREF,
2TMAXO,RMAXM,RMAXO,ZMAXM,ZMAXO,Q4,BIG

```

```

COMMON /FOR/ NN(40,9)

```

```

COMMON /FIV/ NCFI(37),NCFG(37),NNR(37),NRE(37),NNL(37),
1NPN(37)
COMMON /SIX/ IBAND,IEXT,INTP,IVEC,JVEC,KK,LOAD,NCFIT,NCFOT,
1NCHECK,NET,NDF,NGP,NNE,NNLT,NNRT,NPASS,NPROB,NRPPI,NRPPQ,
3NNNT,NNRE,NPNT,IPLANE,NBAND

```

```

ZERO = 0.00
II = IVEC + JVEC + 1
IF (IPLANE.EQ.0) II = II - 1
DO 500 LL = 1, II

```

```

DO 200 I = 1, NDF
F(LL,I) = F(LL,I) / SSM(I,1)
DO 100 J = 2, IBAND
L = I + J - 1
IF (L.GT.NDF) GO TO 200

```



```

100 DUM = SSM(I,J)
200 F(LL,L) = F(LL,L) - DUM * F(LL,I)
    CONTINUE
CCC

F(LL,NDF) = F(LL,NDF) / SSM(NDF,I)
DO 400 L = 2, NDF
K = NDF - L + 1
SUM = ZERO
DO 300 J = 2, IBAND
M = J + K - 1
IF (NDF.LT.M) GO TO 400
SUM = SUM + SSM(K,J) * F(LL,M)
400 F(LL,K) = (F(LL,K) - SUM) / SSM(K,I)
CCC

500 CONTINUE
    CONTINUE

IF (IPLANE.EQ.0) GO TO 20
    CORRECTING THE DISPLACEMENTS

FORSL = 0.00
DO 7 I = 1, NNRE
DUM = 0.00
DO 6 J = 1, NDF
DUM = DUM + COPY(I,J) * F(II,J)
7 FORSL = FORSL + DUM
CCC

IF (JVEC.EQ.0) GO TO 10
    C
DO 8 I = 1, JVEC
SUM = 0.00
DO 9 J = 1, NNRE
K = 2 * NNRE(J)
SUM = SUM + COPY(J,K) * F(IVEC+I,K) * BIG
9 CONTINUE
    C
AXIALF = AXIALF + SUM
8 CONTINUE
10 CONTINUE
    C

```



```

00029440
00029450
00029460
00029470
00029480
00029490
00029500
00029510
00029520
00029530
00029540
00029550
00029560
00029570
00029580
00029590
00029600
00029610
00029620
00029630
00029640
00029650
00029660
00029670
00029680
00029690
00029700
00029710
00029720
00029730
00029740
00029750
00029760
00029770
00029780

```

```

C      DO 11 I = 1, IVEC
C      SUM = 0.0
C      DO 12 J = 1, NNRE
C      K = 2 * NRE(J)
C      SUM = SUM + COPY(J,K) * F(I,K) * BIG
12 CONTINUE
C      FACT = (AXIALF+ SUM)/FORS1
C
C      DO 28 L = 1, NDF
C      F(I,L) = F(I,L) + FACT * F(II,L)
28 CONTINUE
11 CONTINUE
C
C      USING THE PRINCIPLE OF SUPERPOSITION
C
C      20 M = 0
C      IF (LOAD.NE.0) M = M + 1
C      IF (PRES.NE.0) M = M + 1
C      IF (OMEGA.NE.0) M = M + 1
C      IF (CM.EQ.0) GO TO 40
C      DO 41 I = 1, IVEC
C      DO 41 J = 1, NDF
C      DO 41 K = 1, M
C      F(I,J) = F(I,J) + F(IVEC+K,J)
41 CONTINUE
C
C      40 CONTINUE
C      RETURN
C      END

```


00029790

SUBROUTINE STRESS

AT THE TWO GAUSS POINTS ON THE TWO BOUNDARIES OF
EACH ELEMENT (WHERE ETA=+1 AND ETA=-1) STRESSES
ARE CALCULATED IN THIS SUBROUTINE, AND ALSO THE
MAXIMUM MEAN AND OCTAHEDRAL SHEAR STRESSES
ARE EVALUATED AND THEN THE TIME AND LOCATION OF
THE MAXIMUM STRESSES ARE DISTINGUISHED HERE.

IMPLICIT REAL * 8 (A-H,O-Z)

COMMON /ONE/ AJ(2,2),B(4,16),COPY(37,298),D(4,4),DL(4,4),
1DNZR(2,8),ESM(16,16),F(24,298),SSM(298,66),STOR(20,150)

COMMON /TWO/ AL(5),DENS(5),DETA(8),DXI(8),E(5),EPSO(4),
1FE(16),GC(4),GX(4),POI(5),R(149),SHP(8),SHT(5),TK(5),Z(149)

COMMON /TREF/ AXIALF,DTI,EVERY,FORS1,OMEGA,HTCI,HTCO,
1POSI,PRES,Q1,Q2,Q3,SHMAX,SMAX,TIME,TIME1,TINII,TMAXM,TREF,
2TMAXO,RMAXM,RMAXO,ZMAXM,ZMAXO,Q4,BIG

COMMON /FOR/ NN(40,9)

COMMON /FIV/ NCFI(37),NCFO(37),NNR(37),NRE(37),NNL(37),
1NPN(37)

COMMON /SIX/ IBAND,IEXT,INTP,IVEC,JVEC,KK,LOAD,NCFIT,NCFOT,
1NCHECK,NET,NDF,NGP,NNE,NNLT,NNRT,NPASS,NPROB,NRPP1,NRPPQ,
3NNT,NNRE,NPNT,IPLANE,NBAND

DIMENSION EDISP(16),SIG(4),EPS(4)

GX(1) = -.577350269189626
GX(2) = -GX(1)
GC(1) = 1.0D0
GC(2) = GC(1)
PAI = 3.141592653589793
NNE2 = 2 * NNE
DO 25 LL= 1, NET

C

CCCCCCCCCCCCCCCC

C

C

C

C

C

C

C

C


```

C      WRITE (6,200) LL
200  FORMAT (1H1,/,40X,'STRESSES AND TEMPERATURES IN ELEMENT',
C      1I3,/,40X,39('*,',/,40X,39('*,',/))
C      L = LL
C
C      IF (L.EQ.1) GO TO 1
C      IF (NN(L,9).EQ.NN(L-1,9)) GO TO 3
C
C      1 K = NN(L,9)
C      EL = 1.D0 + POI(K)
C      FACT = (E(K)*EL)/((1.D0+POI(K))*(1.D0-2.D0*POI(K)))
C      EL = POI(K) / EL
C
C      D(1,1) = FACT
C      D(1,2) = EL*FACT
C      D(1,3) = D(1,2)
C      D(1,4) = 0.D0
C      D(2,2) = FACT
C      D(2,3) = D(1,2)
C      D(2,4) = 0.D0
C      D(3,3) = FACT
C
C      D(3,4) = 0.D0
C      D(4,4) = E(K)/(2.D0*(1.D0+POI(K)))
C
C      DO 3 I = 1,4
C      DO 3 J = 1,4
C      D(J,I) = D(I,J)
C
C      3 CONTINUE
C
C      ETA = -1.D0
C      DO 9 N = 1, 2
C      ETA = -ETA
C      DO 13 M = 1, 2
C      XI = GX(M)
C
C      DO 7 I = 1,4
C      DO 7 J = 1,NNE2
C      7 B(I,J) = 0.D0

```



```

00030730
00030740
00030750
00030760
00030770
00030780
00030790
00030800
00030810
00030820
00030830
00030840
00030850
00030860
00030870
00030880
00030890
00030900
00030910
00030920
00030930
00030940
00030950
00030960
00030970
00030980
00030990
00031000
00031010
00031020
00031030
00031040
00031050
00031060
00031070
00031080
00031090
00031100
00031110
00031120
00031130
00031140
00031150
00031160
00031170
00031180
00031190
00031200

```

```

SHP(1) = .25D0 * (1.D0-XI) * (1.D0-ETA) * (-XI-ETA-1.D0)
SHP(2) = .5D0 * (1.D0-XI)**2 * (1.D0-ETA)
SHP(3) = .25D0 * (1.D0+XI) * (1.D0-ETA) * (XI-ETA-1.D0)
SHP(4) = .5D0 * (1.D0-ETA)**2 * (1.D0+XI)
SHP(5) = .25D0 * (1.D0+XI) * (1.D0+ETA) * (XI+ETA-1.D0)
SHP(6) = .5D0 * (1.D0+XI)**2 * (1.D0+ETA)
SHP(7) = .25D0 * (1.D0-XI) * (1.D0+ETA) * (-XI+ETA-1.D0)
SHP(8) = .5D0 * (1.D0-ETA)**2 * (1.D0-XI)

```

DERIVATIVES OF THE SHAPE FUNCTIONS WITH RESPECT TO XI.

```

DXI(1) = .25D0 * (1.D0-ETA) * (2.D0*XI+ETA)
DXI(2) = -XI * (1.D0-ETA)
DXI(3) = .25D0 * (1.D0-ETA) * (2.D0*XI-ETA)
DXI(4) = .5D0 * (1.D0-ETA)**2
DXI(5) = .25D0 * (1.D0+ETA) * (2.D0*XI+ETA)
DXI(6) = -XI * (1.D0+ETA)
DXI(7) = .25D0 * (1.D0+ETA) * (2.D0*XI-ETA)
DXI(8) = -.5D0 * (1.D0-ETA)**2

```

DERIVATIVES OF THE SHAPE FUNCTIONS WITH RESPECT TO ETA.

```

DETA(1) = .25D0 * (1.D0-XI) * (2.D0*ETA+XI)
DETA(2) = -.5D0 * (1.D0-XI)**2
DETA(3) = .25D0 * (1.D0+XI) * (2.D0*ETA-XI)
DETA(4) = -ETA * (1.D0+XI)
DETA(5) = .25D0 * (1.D0+XI) * (2.D0*ETA+XI)
DETA(6) = .5D0 * (1.D0-XI)**2
DETA(7) = .25D0 * (1.D0-XI) * (2.D0*ETA-XI)
DETA(8) = -ETA * (1.D0-XI)

```

CONSTRUCTION OF THE JACOBIAN MATRIX.

```

DO 20 I = 1, 2
DO 20 J = 1, 2
20 AJ(I,J) = 0.D0

```

```

DO 22 I = 1, NNE

```



```

C C
AJ(1,1) = AJ(1,1) + DXI(I) * Z(NN(L,I))
AJ(1,2) = AJ(1,2) + DXI(I) * Z(NN(L,I))
AJ(2,1) = AJ(2,1) + DETA(I) * Z(NN(L,I))
AJ(2,2) = AJ(2,2) + DETA(I) * Z(NN(L,I))
22 CONTINUE
C C C C C
DET = AJ(1,1) * AJ(2,2) - AJ(1,2) * AJ(2,1)
DEI = 1.0D0/DET
INVERTING THE JACOBIAN AND STORING IN AJ AGAIN.
C C C C C
DUM = AJ(1,1) * DEI
AJ(1,1) = AJ(2,2) * DEI
AJ(1,2) = -AJ(1,2) * DEI
AJ(2,1) = -AJ(2,1) * DEI
AJ(2,2) = DUM
C C C
CR = 0.D0
DO 2 I = 1,NNE
M1 = NN(L,I)
2 CR = CR + SHP(I) * R(M1)
C C C C C
EVALUATION OF THE B MATRIX
DO 15 I = 1, NNE
K1 = 2 * I - 1
K2 = K1 + 1
B(1,K2) = AJ(1,1) * DXI(I) + AJ(1,2) * DETA(I)
B(2,K1) = AJ(2,1) * DXI(I) + AJ(2,2) * DETA(I)
B(3,K1) = SHP(I) / CR
B(4,K1) = B(1,K2)
B(4,K2) = B(2,K1)
15 CONTINUE
C C C C C
POSITION OF GAUSS POINT
RPOS = 0.D0
ZPOS = 0.D0
DO 8 I = 1, NNE

```



```

C C
      RPOS = RPOS + SHP(I) * R(NN(L,I))
      ZPOS = ZPOS + SHP(I) * Z(NN(L,I))
      WRITE (6,201) XI, ETA, RPOS, ZPOS
201  FORMAT (//,40X,'AT LOCATION XI=',IPD14.7,' AND ETA=',
11PD14.7,'//,53X,'R=',IPD14.7,' AND Z=',IPD14.7,'//,4X,
2,TIME',8X,'SIGMA Z',9X,'SIGMA R',8X,'SIGMA THETA',7X,
3,TAU R Z',7X,'TEMPERATURE',6X,'MEAN STRESS',7X,'OCTA',
4,SHEAR',//)
C
      DO 30 KK = 1, IVEC
C C C C C
          EVALUATION OF THE DISPLACEMENT OF ELEMENT L
          DO 4 I = 1, NNE
          I2 = I * 2
          I1 = I2 - 1
          J2 = NN(L,I) * 2
          J1 = J2 - 1
          EDISP(I1) = F(KK,J1)
          EDISP(I2) = F(KK,J2)
          4
C C C C C
          EVALUATING TEMPERATURE AT GAUSS POINT
          TGP = 0.00
          DO 31 I = 1, NNE
          M1 = NN(L,I)
          31 TGP = TGP + SHP(I) * STOR(KK,M1)
          TGP = TGP - TREF
          TGP = AL(NN(L,9)) * TGP
C
          EVALUATION OF STRAIN VECTOR FOR A GAUSS POINT ON
          THE BOUNDARY OF ELEMENT L.
          DO 5 I = 1, 4
          G0 = 0.00
          DO 5 J = 1, NNE2
          G = G0 + B(I,J) * EDISP(J)
          5 EPS(I) = G
C

```



```

00032170
00032180
00032190
00032200
00032210
00032220
00032230
00032240
00032250
00032260
00032270
00032280
00032290
00032300
00032310
00032320
00032330
00032340
00032350
00032360
00032370
00032380
00032390
00032400
00032410
00032420
00032430
00032440
00032450
00032460
00032470
00032480
00032490
00032500
00032510
00032520
00032530
00032540
00032550
00032560
00032570
00032580
00032590
00032600
00032610
00032620
00032630
00032640
00032650
00032660
00032670
00032680
00032690
00032700
00032710
00032720
00032730
00032740
00032750
00032760
00032770
00032780
00032790
00032800
00032810
00032820
00032830
00032840
00032850
00032860
00032870
00032880
00032890
00032900
00032910
00032920
00032930
00032940
00032950
00032960
00032970
00032980
00032990
00033000
00033010
00033020
00033030
00033040
00033050
00033060
00033070
00033080
00033090
00033100
00033110
00033120
00033130
00033140
00033150
00033160
00033170
00033180
00033190
00033200
00033210
00033220
00033230
00033240
00033250
00033260
00033270
00033280
00033290
00033300
00033310
00033320
00033330
00033340
00033350
00033360
00033370
00033380
00033390
00033400
00033410
00033420
00033430
00033440
00033450
00033460
00033470
00033480
00033490
00033500
00033510
00033520
00033530
00033540
00033550
00033560
00033570
00033580
00033590
00033600
00033610
00033620
00033630
00033640
00033650
00033660
00033670
00033680
00033690
00033700
00033710
00033720
00033730
00033740
00033750
00033760
00033770
00033780
00033790
00033800
00033810
00033820
00033830
00033840
00033850
00033860
00033870
00033880
00033890
00033900
00033910
00033920
00033930
00033940
00033950
00033960
00033970
00033980
00033990
00034000
00034010
00034020
00034030
00034040
00034050
00034060
00034070
00034080
00034090
00034100
00034110
00034120
00034130
00034140
00034150
00034160
00034170
00034180
00034190
00034200
00034210
00034220
00034230
00034240
00034250
00034260
00034270
00034280
00034290
00034300
00034310
00034320
00034330
00034340
00034350
00034360
00034370
00034380
00034390
00034400
00034410
00034420
00034430
00034440
00034450
00034460
00034470
00034480
00034490
00034500
00034510
00034520
00034530
00034540
00034550
00034560
00034570
00034580
00034590
00034600
00034610
00034620
00034630
00034640
00034650
00034660
00034670
00034680
00034690
00034700
00034710
00034720
00034730
00034740
00034750
00034760
00034770
00034780
00034790
00034800
00034810
00034820
00034830
00034840
00034850
00034860
00034870
00034880
00034890
00034900
00034910
00034920
00034930
00034940
00034950
00034960
00034970
00034980
00034990
00035000
00035010
00035020
00035030
00035040
00035050
00035060
00035070
00035080
00035090
00035100
00035110
00035120
00035130
00035140
00035150
00035160
00035170
00035180
00035190
00035200
00035210
00035220
00035230
00035240
00035250
00035260
00035270
00035280
00035290
00035300
00035310
00035320
00035330
00035340
00035350
00035360
00035370
00035380
00035390
00035400
00035410
00035420
00035430
00035440
00035450
00035460
00035470
00035480
00035490
00035500
00035510
00035520
00035530
00035540
00035550
00035560
00035570
00035580
00035590
00035600
00035610
00035620
00035630
00035640
00035650
00035660
00035670
00035680
00035690
00035700
00035710
00035720
00035730
00035740
00035750
00035760
00035770
00035780
00035790
00035800
00035810
00035820
00035830
00035840
00035850
00035860
00035870
00035880
00035890
00035900
00035910
00035920
00035930
00035940
00035950
00035960
00035970
00035980
00035990
00036000
00036010
00036020
00036030
00036040
00036050
00036060
00036070
00036080
00036090
00036100
00036110
00036120
00036130
00036140
00036150
00036160
00036170
00036180
00036190
00036200
00036210
00036220
00036230
00036240
00036250
00036260
00036270
00036280
00036290
00036300
00036310
00036320
00036330
00036340
00036350
00036360
00036370
00036380
00036390
00036400
00036410
00036420
00036430
00036440
00036450
00036460
00036470
00036480
00036490
00036500
00036510
00036520
00036530
00036540
00036550
00036560
00036570
00036580
00036590
00036600
00036610
00036620
00036630
00036640
00036650
00036660
00036670
00036680
00036690
00036700
0003671
```


00032650
00032660
00032670
00032680
00032690
00032700
00032710
00032720
00032730
00032740
00032750
00032760
00032770
00032780
00032790
00032800
00032810
00032820
00032830
00032840
00032850
00032860
00032870
00032880
00032890
00032900
00032910
00032920

```

GO TO 29
27 SMAX = SMEAN
   TMAXM = STOR(KK, NNT+1)
   RMAXM = RPOS
   ZMAXM = ZPOS
29 CONTINUE

      FINDING MAX. OCTAHEDRAL SHEAR STRESS.

      IF (OCTA.GT.SHMAX) GO TO 28
      GO TO 30
28 SHMAX = OCTA
   TMAXO = STOR(KK, NNT+1)
   RMAXO = RPOS
   ZMAXO = ZPOS

      CONTINUE
13 CONTINUE
25 CONTINUE

      RETURN
      END

```


APPENDIX C
USER'S MANUAL

In this section the procedure for the use of the present computer program is described. It is intended that a person with minimum familiarity with the details of the finite element method and computer programming be able to use this program.

The steps below are in order and the user is advised to follow them carefully.

For finding the thermal stresses in an axisymmetric body under axisymmetric loading, the user has the choice of using either the Metric or British system of units. The program will handle both systems and the necessary conversions are made automatically within the program. However, the units used in each system must be consistent and they should be as listed below in Table V.

Now for preparation of the data input for a given axisymmetric geometry with prescribed thermal and structural boundary conditions we go through the details with a simple example.

Step 1:

Draw the longitudinal cross-section of the body to scale. Identify the cylindrical coordinates R and Z , with the origin of the Z axis on the most left-hand point of the

TABLE V
UNITS FOR INPUT DATA

Note: an input card specifies whether British or Metric data is being used.

Quantity	British Unit Syst.	Metric Syst.
Coordinates	in.	cm
Time	sec.	sec.
Temperature	°F	°C
Velocity	ft./sec.	m/sec.
Axial force [✓]	lbf	kg
Pressure	lbf/in ²	kg/cm ²
Density	lbm/ft ³	gm/cm ³
Specific heat	Btu/lbm°F	cal/kg°C
Mod. of elasticity	lbf/in ²	kg/mm ²
Coef. of thermal exp.	1/°F	1/°C
Thermal conductivity	Btu/hr.ft.°F	cal/sec.cm.°C
Heat transfer coef.	Btu/hr.ft. ² °F	cal/sec.cm. ² °C
Load vector	lbf	kg
Rotational speed	Revolutions/min.	Revolutions/min.

[✓]lbf is pound force and lbm is pound mass.

body. The entire cross section must lie in the first quadrant of the coordinate plane.

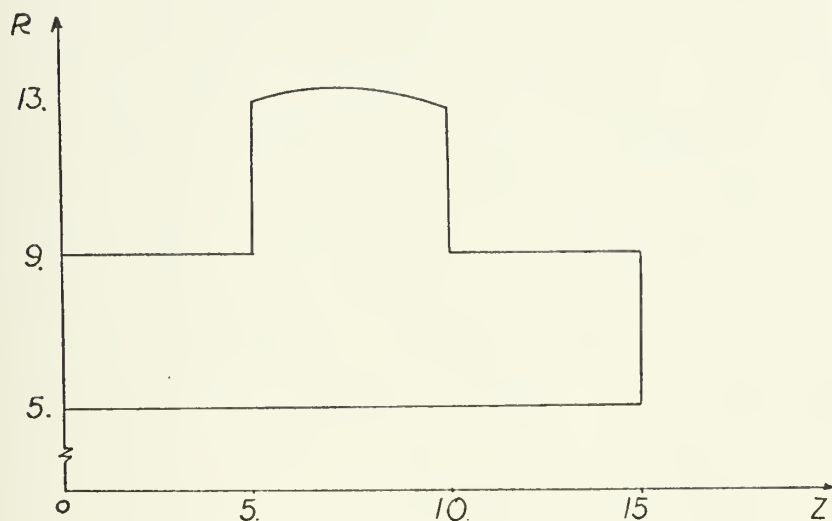


Figure 14. Longitudinal Cross Section.

Step 2:

Subdivide the cross section into a maximum of 40 quadrilateral elements. This subdivision is extremely important and we should provide smaller elements for the parts of the cross section where we expect the largest stresses or temperature gradient. If the body is made from several materials, elements should be chosen so that each element contains only one material. Any two adjacent elements must share one complete side of the quadrilateral. Number the elements, starting from 1, in any arbitrary manner (see Fig. 15).

Step 3:

Since eight-noded elements are used in the program, identify these nodal points around the boundary of each element. Four nodes will be at the corners and the other four will be

at the mid-points of the sides. Number sequentially these nodes starting from 1 and increasing in the direction where the number of elements is the least. (The numbers thus assigned are called global node numbers.) For clarity of this step assume a cross-section as in Figure 15 such that the maximum number of elements in one direction is less than the maximum number of elements in other direction. (In Fig. 15 we have maxima of 2 and 3 elements in R and Z directions respectively.) Thus we number the nodes beginning in the R direction. See Fig. 16. This method will give the minimum band width of the stiffness matrix and will save computer execution time.

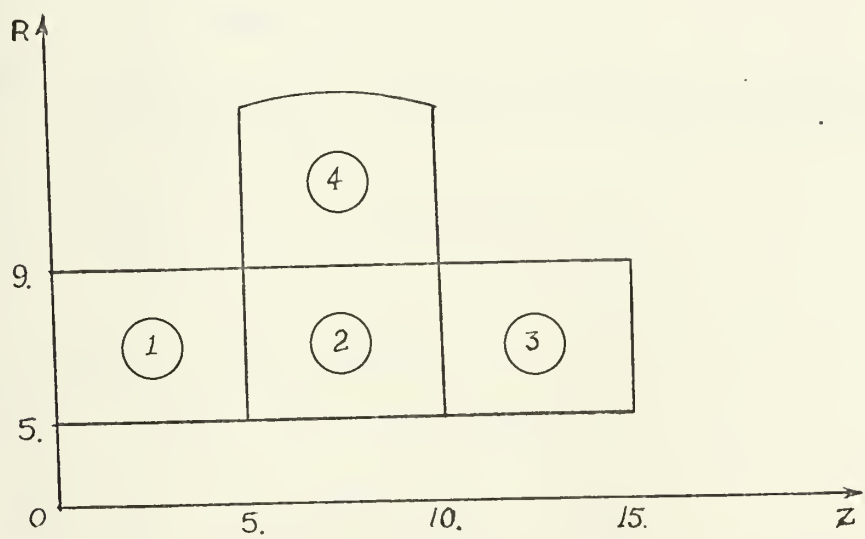


Figure 15. Subdivision Inot Elements.

Step 4:

At this point we are ready to prepare the first data card. Input quantities are:

On this card, or any succeeding one, if an input quantity (e.g., NØFN) is zero, the corresponding field may have a 0 or be left blank, unless otherwise stated.

Step 5:

For each corner node a card should be punched which reads: The node number I, the R coordinate and the Z coordinate of the node. The format is (I5, 2F10.5).

It is not necessary to sort these cards in the order of increasing or decreasing corner node numbers, they can be put together in any arbitrary order. A typical card is illustrated in step 7. There must be as many punched cards as the number of corner nodes (NCN).

Step 6:

In this step we read in the connectivity array, i.e., for each element we prepare one card which gives the global node numbers and the material identification number.

For each element, start from the lower left-hand node and move in the counterclockwise direction within that element and punch the global node numbers in order. The format is (9I5) where the last I5 is the material identification number.

It is very important that the cards prepared for this step be put together in order of elements, i.e., the first card for element 1, the second card for element 2, etc. For ease of sorting the connectivity cards the element number may be punched after column 50. As an example, for element 2 of Fig. 16 the connectivity card should read:

[illegible]

Step 7:

For each mid-side node that is on a curved element edge a card is prepared with format (I5,2F10.5) to read the global node number and the R and Z coordinates. Here, as in step 5, these cards may be put together in any arbitrary order. For the case of Fig.16, there will be only one card to be punched and it would read:

[illegible]

141

Step 8:

The properties of the different materials are specified in this step. For each material a card must be prepared that gives the modulus of elasticity (E), coefficient of thermal expansion (AL), Poisson's ratio (PØI), thermal conductivity (TK), density (DENS) and specific heat (SHT).

I.e., read:

E, AL, PØI, TK, DENS, SHT

with the format (2D12.4, 4F8.3). The first card will be for material number 1, the second card for material number 2, etc.

Step 9:

For this step a single card must be punched, starting in column one, which reads the word BRITISH or METRIC in accordance with the system of units used.

This completes the input of geometric and material property data.

Step 10:

For a transient temperature problem only (no stress calculations) leave a blank card for this step and proceed directly to step 16.

For a stress problem or thermal stress problem, in this step we specify the type of problem and the structural boundary conditions.

The following quantities, when pertinent, are to be specified.

ØMEGA	The speed of rotation, about the Z axis, in revolutions per minute
PRES	The external pressure applied to a boundary segment
IEXT	The number of known temperature vectors for which evaluation of thermal stresses is desired
LØAD	The indication for an additional known load vector. If there is one, LØAD = 1, otherwise LØAD = 0
NNLT	Total number of nodes fixed in the longitudinal direction
NNRT	Total number of nodes fixed in the radial direction
IPLANE	An indication for the plane-end boundary condition. If desired IPLANE = 1

If any of these items is not applicable, the corresponding field is left blank.

Now we have to punch a single card for this step which reads:

ØMEGA, PRES, IEXT, LØAD, NNLT, NNRT, IPLANE

with format (2F10.4, 5I5).

Step 11:

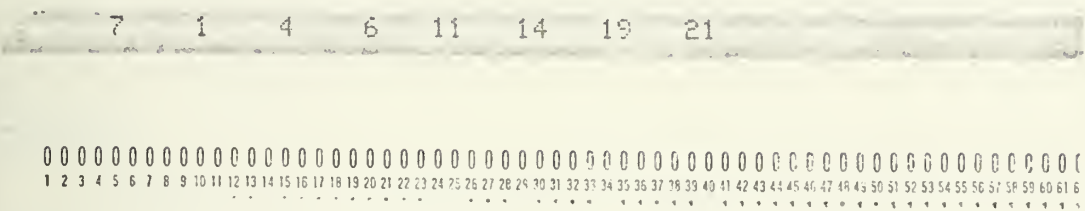
If there is no pressure loading (PRES=0), omit this step. For non-zero pressure, the total number of nodes on pressure loading boundary segment (NPNT) and the global node numbers of these (pressure) nodes (NPN(I)) must be specified here. Only one segment is permitted.

We read in:

NPNT, (NPN(I), I=1, NPNT)

with the format (12I5).

In our example, Fig. 16, if there is pressure inside the body, then the data card for this step will be:



Step 12:

If there is no additional load vector ($L\emptyset AD = 0$), omit this step.

For $L\emptyset AD = 1$ read in the additional load vector with the format (6F10.4). The components of the load vector are arranged in the order: R component at node 1, Z component at node 1, R component at node 2, etc. I.e., READ (F(I),I=1,NDF), where NDF (the number of degrees of freedom) is equal to two times the total number of nodes ($2*NNT$).

Step 13:

In any stress problem, or thermal stress problem, there must be at least one node constrained against longitudinal motion, i.e., $NNLT > 0$, so we must specify here which nodes are to be constrained against longitudinal motion. These global node numbers $NNL(I)$ are read in with the format (I0I5), i.e.,

```
READ: (NN(I),I=1,NNLT)
```

In our example, if global nodes 1 and 21 are fixed in the Z direction, we have a card that reads:

If there are no nodes fixed in the radial direction (NNRT = 0), omit this step.

For $NNRT > 0$ we must read in the global numbers (NNR(I)) of those nodes which are to be fixed against radial motion. The format is (10I5), i.e.,

READ: (NNR(I), I=1, NNRT)

If node 1 and 21 of our example are also fixed in the R direction, then the card for this step would read exactly as the one in Step 13.

If there is no plane-end boundary condition (IPLANE = 0), omit this step.

For the case of end-planes-remain-plane boundary condition, i.e., (IPLANE = 1), we have to specify the total number of nodes of the right-hand end (NNRE) and the global node numbers of this end (NNR(I)).

We read in:

$$\text{NNRE}, (\text{NNR}(I), I=1, \text{NNRE})$$

with the format (10I5).

For the case of Fig. 16, if it is desired to have end-planes-remain-plane boundary condition then the data for this step will be:

Step 16:

In this step we begin the specification of thermal boundary conditions. Additional details are prescribed in Steps 19 through 24.

Thermal boundary conditions are imposed along discrete segments of the boundary. Each such segment must begin and end at a corner node of a boundary element. We consider separately the imposition of the various kinds of thermal boundary conditions.

(a) Convection

Since the fluid temperature for convection is determined from a prescribed temperature-time history at entry section and a specified flow velocity (see Appendix D, Section 5), it is necessary that the terms "inside," i.e., adjacent to the symmetry axis, and "outside" have their ordinary meanings. Thus on the inside surface the convection boundary condition may be applied to a single (continuous) segment. A similar prescription may be employed for the outside portion. The entry section used for flow calculations is at the upstream end of corresponding segment.

(b) Constant temperature

To specify constant temperature on a portion of the boundary, the designator "inside" or "outside" may be used. Such a constant temperature portion may consist of several

discrete segments. If two different constant temperature portions are prescribed, one may be designated "inside" and the other "outside."

(c) Combinations of convection and constant temperature

The convection and constant temperature conditions can be used together, but the portion designated "inside" must have only one of these conditions prescribed and the same restriction applies to the "outside."

(d) Insulated

All portions of the boundary not included in the segments specifically identified as "inside" and "outside" are considered insulated.

The following items, when pertinent, are to be given as specified below:

TINIT	The constant initial body temperature
$Q2=0.$	For insulated boundary condition outside
$Q2 > 0.$	For convection or constant temperature boundary condition outside
$Q3=0.$	For insulated boundary condition inside
$Q3 > 0.$	For convection or constant temperature boundary condition inside
$Q4 < 0.$	If solving stress problem only
$Q4 > 0.$	If solving temperature problem only
$Q4=0.$	If solving thermal stress problem
AXIALF	The axial force in the Z direction. As usual, + (plus) for tension and - (minus) for compression.

So, we read:

TINIT, Q2, Q3, Q4, AXIALF

with the format (5F10.4).

In case of Fig. 16, if the initial solid temperature is 60° and we have insulated outside and convection boundary condition inside with 120 kg axial compressive force, then the card for this step will be:

120.00

[illegible]

where the blanks left in the columns 31 through 40 indicate that we want to solve a thermal stress problem.

Step 17:

No action is taken in this step - we merely choose between proceeding to step 18 or jumping to step 26.

If there are no known temperature vectors for which evaluation of thermal stresses is desired, proceed to step 18.

For IEXT > 0, i.e., when some known temperature vectors are to be entered for thermal stress analysis alone or combined with some other loadings, proceed directly to step 26.

Step 18:

If ($Q4 < 0$), i.e., we are to solve only a stress problem, proceed directly to step 27.

Step 19:

If there is an insulated boundary condition outside ($Q2 = 0$), omit the following steps and start from step 22.

For the case of a convection or constant temperature boundary condition outside ($Q2 > 0.$), we have to specify the outside heat transfer coefficient ($HTC\emptyset$)(for constant temperature $HTC\emptyset = 10^{20}.$), the constant outside initial fluid temperature ($TEMP\emptyset$) which may be equal to the solid's initial temperature, the total number of nodes in contact with fluid outside ($NCF\emptyset T$), and the number of temperature ramps for outside flow ($NRAMP\emptyset$). We read:

$HTC\emptyset, TEMP\emptyset, NCF\emptyset T, NRAMP\emptyset$

with the format (D16.4,F10.4,2I5). See example in step 22.

Step 20:

Here we read in the global node numbers of the nodes in contact with outside fluid ($NCF\emptyset(I)$). The sequence of these node numbers must be in the direction of flow velocity. Read:

$(NCF\emptyset(I), I=1, NCF\emptyset T)$

with the format (12I5) (see example in step 23).

Step 21:

For each ramp of outside flow we read in the time when the ramp starts $BDRY\emptyset(I,1)$, the initial temperature of the ramp $BDRY\emptyset(I,2)$, the final temperature of the ramp (specify only if it differs from the initial temperature of the next

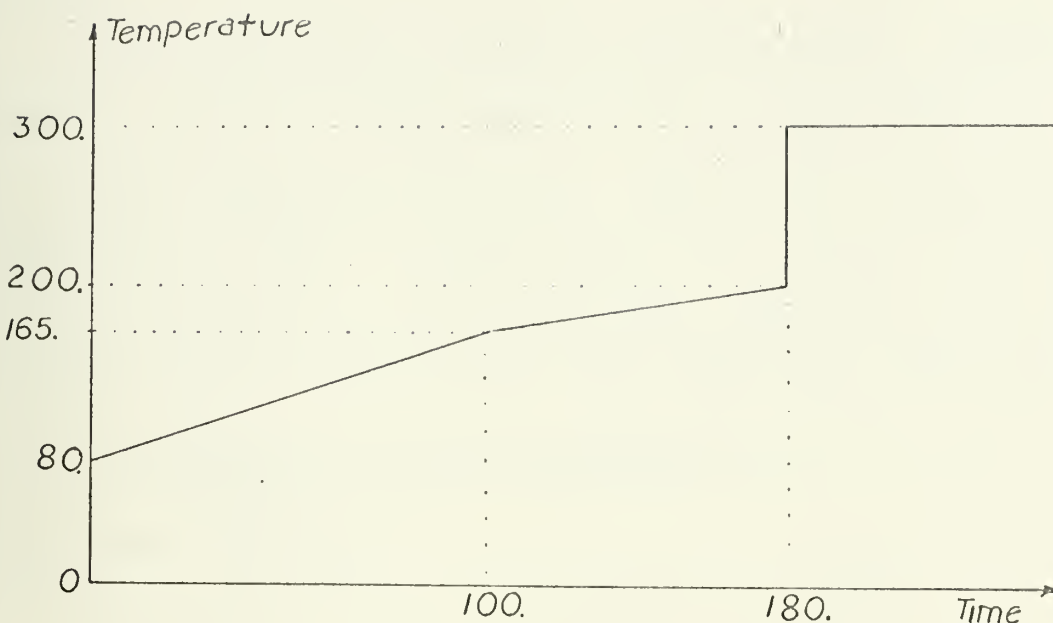


Figure 17. Entry Temperature-Time Variation.

Further details of this temperature-time history are specified in step 24.

Step 23:

Here we read in the global numbers of those nodes which are in contact with inside fluid, (NCFI(I)) with the format (12I5). It is important to read these node numbers in the direction of flow velocity.

For example, if the inside flow of Fig. 16 is from left to right, then for this step the card reads:

[illegible]

Step 24:

In this step data for inside flow are given. For each ramp I we read: the time when the ramp starts BDRYI(I,1), the initial temperature of ramp I BDRYI(I,2), the final temperature of ramp I (specify only if it differs from the initial temperature of the next ramp) BDRYI(I,3), and the velocity of the fluid for this ramp BDRYI(I,4). We read in:

$$((\text{BDRYI}(I,J), J=1,4), I=1, \text{NRAMPI})$$

with format (4F10.4).

For example of Fig. 17, let the flow velocity for the first ramp be 15 and that for the second and third segment be 20. Three cards are needed:

180.00	300.00	300.00	20.00
--------	--------	--------	-------

3^{ed} CARD

0000 00 0000 00 00000 00 000000 00 0000000000000000
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52

100.D0 165.D0 200.D0 20.D0

2nd CARD

[illegible]1st CARD[illegible]

Step 25:

In this step we must specify the following items:

DTI	The time increment (step size) of trapezoidal integration
-----	---

TIME1 The time when the first calculated temperature vector is to be stored for thermal stress evaluation

Step 26:

If there are no known temperature vectors ($IEXT = 0$) for which thermal stress evaluation is desired, omit this step.

For $IEXT > 0$ we must specify the nodal temperatures which are to be stored for thermal stress evaluation. We read:

((STØR (I,J) , J=1,NNT), I=1,IEXT)

with the format (6F10.4).

This means that we enter all the components of the first nodal temperature vector, followed by the components of the second nodal temperature vector and so on until $IEXT$ such vectors have been entered.

Step 27:

The data cards for this problem are completed now. If no more problems are to be solved for the same geometry, omit this step.

If another problem is to be solved for the same geometry and spatial discretization, increment by 1 the NPRØB in step 4 and start the input of the new problem from step 10.

Step 28:

If there is no access to the IBM 360 computer of the Naval Postgraduate School, omit the following steps and start from step 32.

For the convenience of the user, the author has put a so-called CHECK program and the program presented in

Appendix B in the computer system at N.P.S. (Naval Post-graduate School). It is advised, however, that the user check his input data deck with the CHECK program before attempting to solve the problem.

For using the CHECK program prepare the following control cards.

```
//XXXX0000 JØB (0000,0000FT,XX00),'NAME',TIME=1
//JØBLIB DD DSN=F0609.BAKH,DISP=SHR,UNIT=2314,VØL=SER=DUFFY
//GØ EXEC PGM=CHECK,REGION=100K
//FT06F001 DD SYSØUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=3325),
//          UNIT=SYSØUT
//FT05F001 DD *
```

where the first card is the regular FORTRAN job card used at this institution.

Prepare your deck as Fig. 18 and it may be read in from the hot card reader.

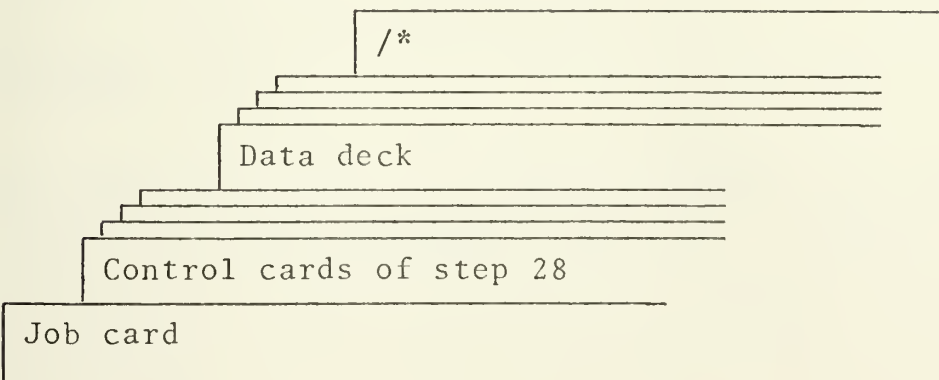


Figure 18. Set-up for Using CHECK Program.

If the output of the CHECK program has any error messages or has not been run, then there are some mistakes in the data deck or control cards.

If there are no error messages on the output of CHECK program, then study carefully the output and make sure it is the same problem that has been intended to be solved. Once the correctness of the input data has been insured, proceed to the next step.

Step 29:

AXITTS stands for Axisymmetric Transient Thermal Stress and this is the name given to the program presented in Appendix B when stored in the computer. For using that we must prepare the following control cards.

```
//XXXX0000 JØB (0000,000FT,XX00)'NAME',TIME=10
//JØBLIB DD DSN=F0609.BAKH,DISP=SHR,UNIT=2314,VØL=SER=DUFFY
//GØ EXEC PGM=AXITTS,REGION=425K
//FT06F001 DD SYSØUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=3325),
// UNIT=SYSØUT,SPACE=(CYL,(6,1))
//FT05F001 DD *
```

where the first card is the regular FØRTRAN job card. It is advised to ask for 10 minutes time, i.e., TIME=10, since this would not affect the priority of the job within the class K jobs.

Prepare the deck as in Fig. 19 (it may be read into the computer from the hot card reader) and submit a so-called

service request card, available in the computer center, to the operator on duty.

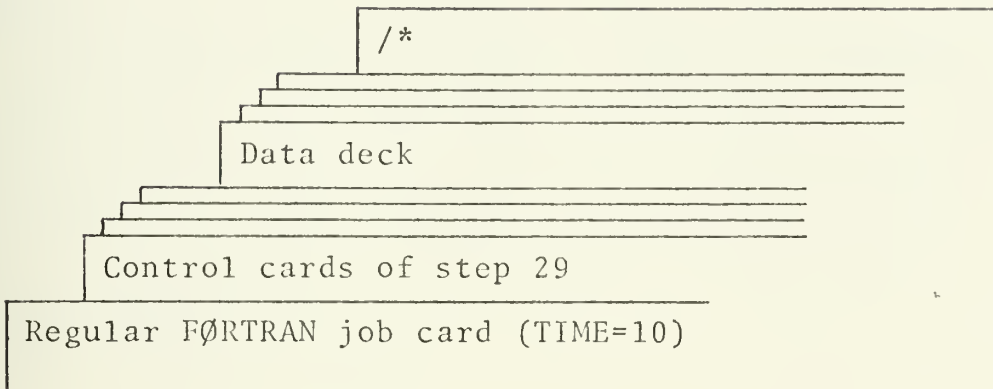


Figure 19. Deck Set-up for Using AXITTS Program.

Step 30:

If the user wishes to have a listing of the program he may prepare the following control cards as in Fig. 20 since the program is also listed on the data cell for this purpose.

Regular FORTRAN job card

```
//PRNT EXEC PGM=IEBPTCH
```

```
//SYSPRINT DD DUMMY
```

```
//SYSUT1 DD DSN=F0609.BAKH,DISP=OLD,UNIT=2321,
```

```
// VOL=SER=CEL003,DCB=(RECFM=FB,LRECL=80,BLKSIZE=2000)
```

```
//SYSUT2 DD SYSOUT=A,SPACE=(CYL,6)
```

```
//SYSIN DD *
```

```
PRINT TYPORG=PO,MAXFLDS=1,MAXNAME=1
```

```
MEMBER NAME=AXITTS
```

```
RECORD FIELD=(80)
```

```
/*
```

Figure 20. Deck Set-up for Obtaining a Listing of the Program AXITTS.

The set-up deck of Fig. 20 may be read in from the hot card reader to get a listing of the program.

Step 31:

If the user wishes to obtain a deck of the program AXITTS he may prepare a deck as in Fig. 21 and read it in from the hot card reader. He must notify the operator on duty to be prepared for punching almost two boxes of IBM cards.

Regular FØRTRAN job card

```
//PUNCH EXEC PGM=IEBPTCH
//SYSPRINT DD DUMMY
//SYSUT1 DD DSN=F-0909.BAKH,DISP=ØLD,UNIT=2321,
// VØL=SER=CEL003,DCB=(RECFM=FB,LRECL=80,BLKSIZE=2000)
//SYSUT2 DD SYSØUT=B,SPACE=(CYL,6)
//SYSIN DD *
        PUNCH TYPØRG=PØ,MAXFLDS=1,MAXNAME=1
MEMBER NAME=AXITTS
RECØRD FIELD=(80)
/*
```

Figure 21. Deck Set-up for Obtaining a Punched Deck of the Program AXITTS

Step 32:

For using the program presented in Appendix B, if the user does not have access to the computer facility at N.P.S., he must punch a copy of the program. (Good luck!)

For the storage location 425K bytes are required and since the output may be longer than what usually is allowed, three additional cylinders are recommended. The execution time required depends on the size of the problem and the number of time integration steps; however, 10 minutes computer time would be sufficient for a problem of 140 nodes and 500 steps of time integration, with 20 temperature vectors stored for stress calculation.

APPENDIX D

PROGRAMMING

The computer program presented in Appendix B is formed from one main program and fifteen different subroutines. The communication between the main program and the subroutines is handled through the common blocks. In order to minimize storage requirements, most of the storage location of the system stiffness matrix is used for temperature calculations. This is accomplished by use of EQUIVALENCE statements. The total storage requirement is 425K bytes.

In this section the function of some parts of the program is discussed and the assumptions used are brought to attention.

1. MAIN PROGRAM

The main program is simply calling different subroutines when they are needed. The subroutines which are called in main program are:

INPUT, PRØB, CANDY, FLØW, FØRMV, TEMPER, PLØT, STIFF, FØRMF, CENTF, PRESS, DISPL, AND STRESS.

2. SUBROUTINE INPUT

In this subroutine the data regarding the geometry of the body, subdivisions into elements, and the material properties are read in and printed out. Calculation of the mid-side coordinates based on the straight line is done

here. This subroutine calls for subroutine PLØTRZ once to produce a graphical representation of the nodal points then the half-band-width of the system stiffness matrix is calculated from the connectivity array. Also, depending upon the choice of the system of units, the necessary conversions in each system of units are accomplished and the reference temperature (70°F or 20°C) based on the system of units is selected here.

3. SUBROUTINE PRØB

For each problem this subroutine is called once by the main program to read in and print out the information regarding the nature of the problem. Based upon the given information the type of the problem is distinguished here and for the thermal problems the length of time integration is calculated.

4. SUBROUTINE CANDY

Subroutine CANDY (C and Y) evaluates the capacitance matrix $\underline{\underline{C}}$ and admittance matrix $\underline{\underline{Y}}^+$ of Eq. 13, in banded form. For each problem this subroutine is called once. The functional flow chart of the subroutine CANDY is given in Fig. 22. Since the only non-zero elements of the symmetric matrix $\underline{\underline{Y}}^*$ (Eq. 11) are the diagonal, first and second super-diagonals, in order to conserve storage and reduce the number of arithmetic operations, the non-zero elements are stored in three different vectors (DIAG, ØFF1, ØFF2).

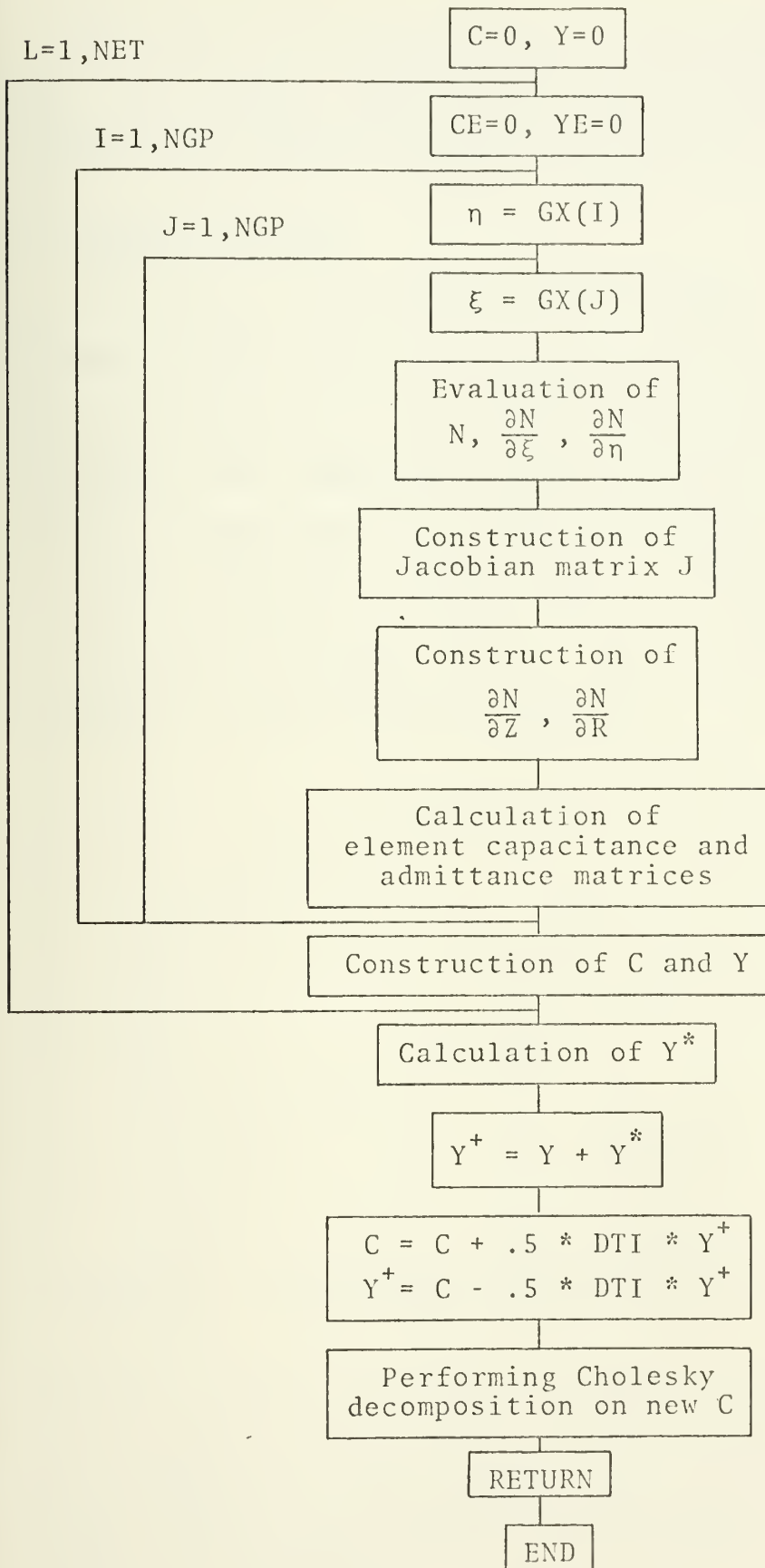


Figure 22. Functional Flow Chart of CANDY.

Once the system capacitance and admittance matrices $\underline{\underline{C}}$ and $\underline{\underline{Y}}^+$ are formed, then the matrices $\underline{\underline{A}}$ and $\underline{\underline{G}}$ (Eq. 29) are evaluated and replace $\underline{\underline{C}}$ and $\underline{\underline{Y}}$ respectively.

5. SUBROUTINE FLØW

If there is any convection or constant temperature thermal boundary condition, this subroutine is called from the main program for each step of time integration in order to evaluate the temperature of the fluid nodes. The calculation is based upon the constant fluid flux assumption (for both inside and outside flow). Consider a section of an irregular cylindrical pipe as Fig. 23 and focus attention on element $i+1$ on the inner boundary.

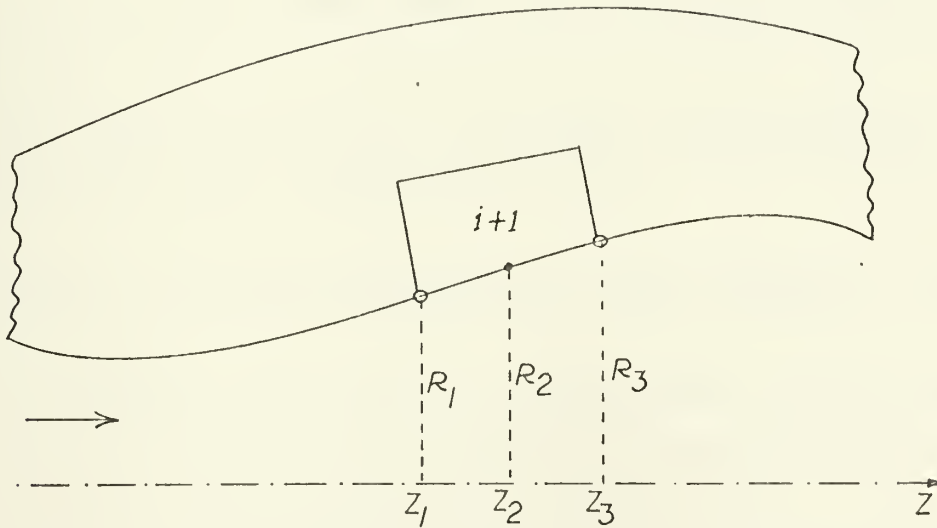


Figure 23. Fluid Node Representation for Inside Flow.

If fluid is moving from left to right we may number the R and Z coordinates of the fluid nodes of the element $i+1$ as in Fig. 23, then the volume of fluid pumped into the pipe at time τ is given by

$$Q_{in} = a v \tau, \quad (56)$$

where a and v are, respectively, the area and velocity at the entrance section. Also the volume of the pipe up to the mid-node of element $i+1$ is

$$Q_1 = V_i + \int_{Z_1}^{Z_2} \pi R^2 dZ. \quad (57)$$

where V_i is the volume of the pipe between the entrance section and element $i+1$ on the inner boundary.

Now we can write

$$R = \sum_{i=1}^3 R_i N_i, \quad (58)$$

where N_i are the one dimensional shape functions (given in Appendix A).

If we substitute Eq. 58 into 57 and make the coordinate transformation, after integration we get

$$Q_1 = V_i + \frac{\pi}{120} (Z_2 - Z_1) (31R_1^2 + 64R_2^2 + R_3^2 + 46R_1R_2 - 8R_1R_3 - 14R_2R_3). \quad (59)$$

Now by equating Eq. 59 to Eq. 56 at any time τ we may determine whether the front of the flow has already passed the mid-node of the wetted side of the element $i+1$.

A similar argument can be carried out for the end node of the element $i+1$. In that case we also get an equation similar to Eq. 59 with different numerical coefficients.

For the case of the outside flow it has also been assumed to have a constant fluid flux and a fictitious

entrance circular cross-sectional area (whose radius is the largest R plus unity) has been assumed.

The numerical coefficients of R 's in Eq. 59 are used in subroutine FLØW and it has been assumed that the temperature of a fluid "particle" does not change during passage through the active section. This is believed to be an acceptable approximation for representative values of flow velocity and active length. With this assumption, for a given entry time-temperature relation (inside and outside flow) this subroutine evaluates the fluid nodal temperatures at any time.

6. SUBRØUTINE FØRMV

At every step of time integration this subroutine is called by the main program to form the vector \underline{y} of the right-hand side of the discretized finite element (Eq. 13) for the given thermal boundary conditions. The program itself is self explanatory.

7. SUBRØUTINE TEMPER

The nodal temperatures are calculated in this subroutine with the trapezoidal time integrations. Irons' correction is applied here after every 10 steps of time integration. The temperature vectors selected for the stress analysis are stored. The transient temperatures will be printed out at the desired interval of time.

8. SUBROUTINE STIFF

For any stress or thermal stress problem subroutine STIFF is called once by the main program to evaluate the stiffness matrix of the system. Once the stiffness matrix of the system is calculated then the desired structural boundary conditions are applied and, at the end, the Cholesky decomposition is performed. The functional flow chart of subroutine STIFF is given in Fig. 24.

9. FØRMF

In subroutine FØRMF the thermal load vectors are calculated for as many as (IVEC) given temperature vectors. These thermal load vectors are the columns of a rectangular matrix F. The provision is made that no matter how many temperature vectors are given (always $IVEC \leq 20$) the IVEC number of columns of the F are filled with the thermal load vectors and the last column of F will contain the load vector corresponding to the unit end displacement for zero axial force when the plane-end boundary condition is applied. The flow chart of subroutine FØRMF is given in Fig. 25.

10. SUBROUTINE CENTF

If the system is rotating about the axis of revolution, this subroutine is called once by the main program to evaluate the centrifugal load vector. This load vector is always placed in the first column after the thermal load vectors (IVEC+1 position) in F.

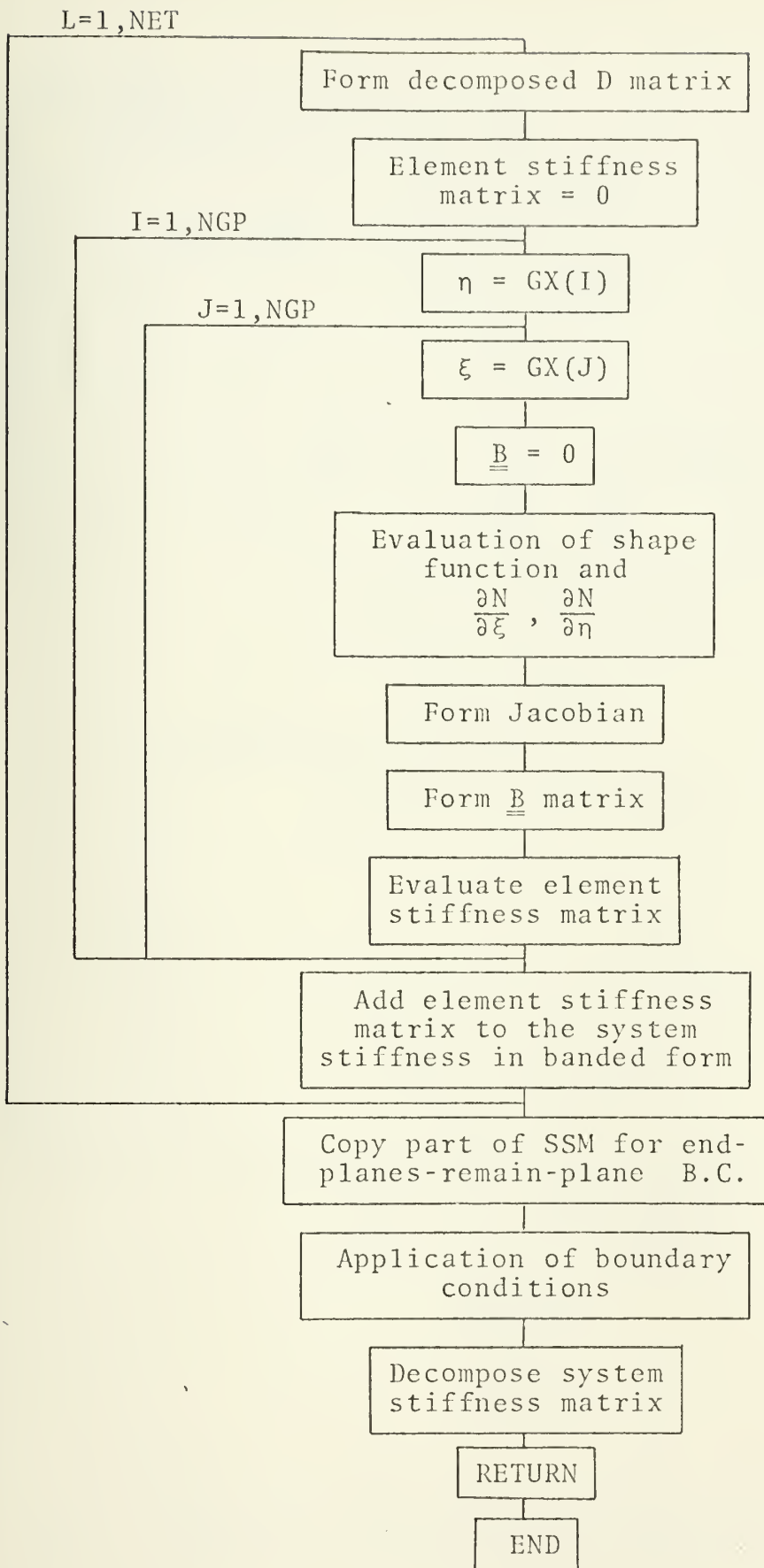


Figure 24. Functional Flow Chart of STIFF.

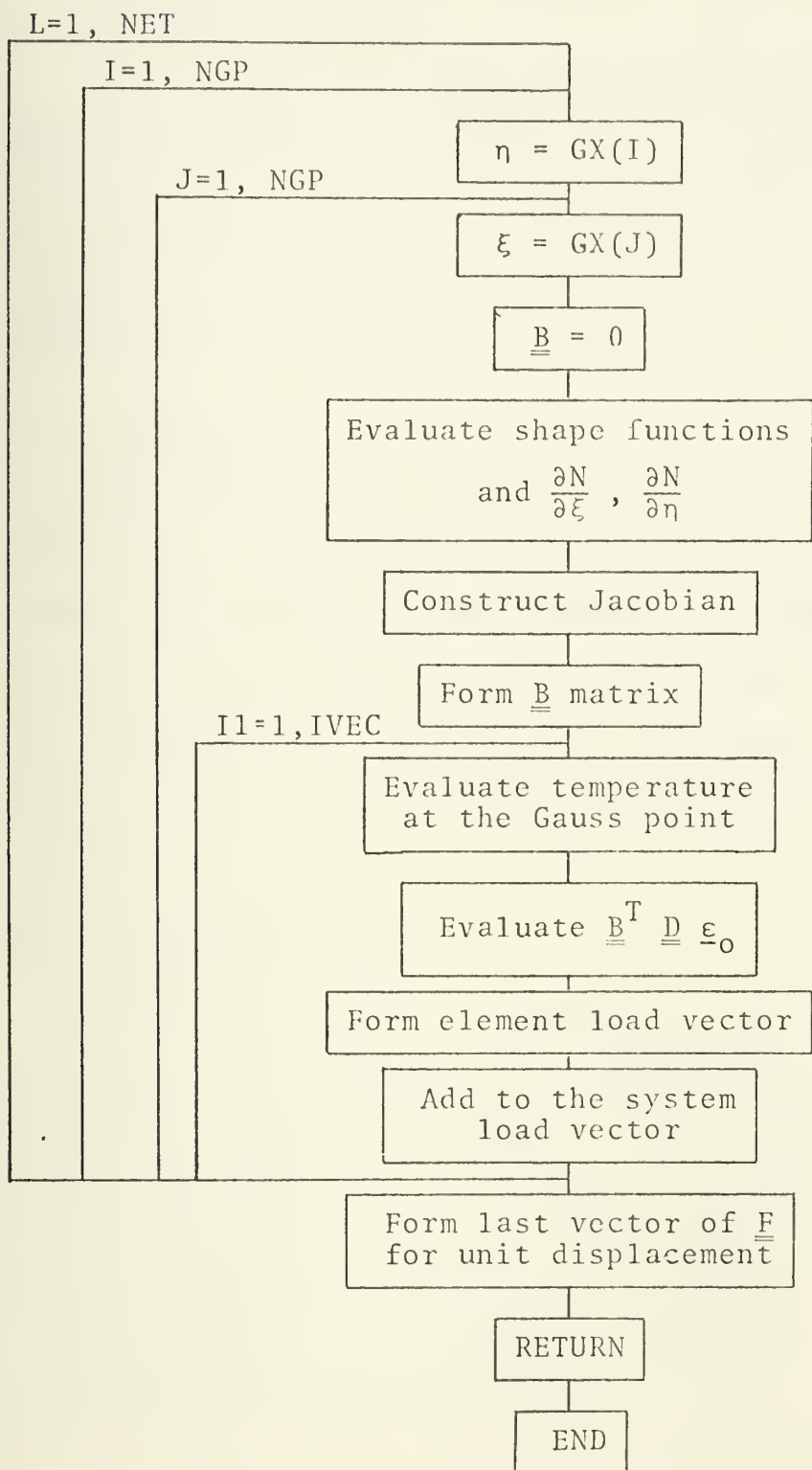


Figure .25. Functional Flow Chart of FØRMF.

11. SUBROUTINE PRESS

If there is any pressure acting on the system, the pressure load vector is calculated in this subroutine and this vector is placed in the column next to centrifugal load vector, if any, otherwise it will fill the position designated for centrifugal load vector in F.

12. SUBROUTINE DISPL

The displacement vectors are found in this subroutine. Since the stiffness matrix is already decomposed in banded form, then the displacement vectors one after another are obtained by back and forward substitution. The principle of superposition is applied here and finally the axial force, if any, is corrected for the plane-end boundary condition.

13. SUBROUTINE STRESS

For every problem this subroutine is called by the main program once to evaluate the stresses at the points $(\eta=\pm 1, \xi=\pm \frac{1}{\sqrt{3}})$ of each element. The transient stresses are calculated and printed for each displacement vector. For each problem the maximum mean stress and the maximum octahedral shearing stress, together with the corresponding times and locations, are printed.

14. LIMITATIONS

In the process of the development of the program discussed so far, it has been intended that all of the calculations and storage of data occur in the computer without

using any external devices such as disks or magnetic tapes so as to be able to solve any sizable problem in relatively short time.

For this reason the following limitations (Table VI) are set forth which give an overall size of 450K bytes to the program.

TABLE VI
MAXIMUM VALUES FOR PROGRAM PARAMETERS

NBAND	Half band-width of the system stiffness matrix	66
NMAT	Total number of different materials	5
NET	Total number of elements	40
NNT	Total number of nodes	149
NCFØT	Total number of nodes in contact with outside fluid	37
NCFIT	Total number of nodes in contact with inside fluid	37
NNRE	Total number of nodes of right-hand end	37
NNLT	Total number of nodes constrained against any longitudinal motion	37
NNRT	Total number of nodes constrained against any radial motion	37
IVEC	Total number of temperature vectors stored for thermal stresses	20
NRAMPØ	Total number of ramps for outside flow	15
NRAMPI	Total number of ramps for inside flow	15
NPNT	Total number of pressure nodes	37

LIST OF REFERENCES

1. Zienkiewicz, O. C., The Finite Element Method in Engineering Science. McGraw-Hill, 1971.
2. Arpaci, Vedat S., Conduction Heat Transfer, Addison-Wesley Publishing Company, 1966.
3. Collatz, Lothar, Numerical Treatment of Differential Equations. Third edition. Springer-Verlag, New York Inc., 1966.
4. Private communication from Bruce M. Irons to Robert E. Newton.
5. Chapman, J., Heat Transfer, Second edition. MacMillan Company, 1967.
6. Timoshenko, S. and Goodier, J. N., Theory of Elasticity, Second Edition, McGraw-Hill, 1951.
7. Fung, Y. C. Foundations of Solid Mechanics. Prentice-Hall, 1965.
8. Irons, B. M., International Journal for Numerical Methods in Engineering, Vol. 2, No. 1, p. 5-35, January-March 1970.
9. Boley, Bruno A. and Weiner, Jerome H., Theory of Thermal Stresses. John Wiley and Sons, Inc., 1962.

INITIAL DISTRIBUTION LIST

	No. Copies
1. Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2. Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3. Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	2
4. Imperial Iranian Naval Headquarters Old Shemiran Road Teheran, Iran	1
5. University of Teheran Department of Mechanical Engineering Shah Reza Avenue Teheran, Iran	1
6. Prof. Robert E. Newton (Code 59Ne) Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	5
7. Prof. John E. Brock (Code 59Bc) Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	1
8. Prof. Paul Pucci (Code 59Pc) Department Of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	1
9. Prof. Gilles Cantin (Code 59Ci) Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	1
10. Lt. M. Bakhshandehpour Zarrab Khaneh Abhar St. No 44 Teheran, Iran	2

UNCLASSIFIED

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

ORIGINATING ACTIVITY (Corporate author)

Naval Postgraduate School
Monterey, California 93940

2a. REPORT SECURITY CLASSIFICATION

Unclassified

2b. GROUP

REPORT TITLE

FINITE ELEMENT SOLUTION FOR AXISYMMETRIC TRANSIENT THERMAL STRESSES

DESCRIPTIVE NOTES (Type of report and inclusive dates)

Mechanical Engineer's Thesis; June 1972

AUTHOR(S) (First name, middle initial, last name)

Manouchehr Bakhshandehpour; Lieutenant, Imperial Iranian Navy

REPORT DATE

June 1972

7a. TOTAL NO. OF PAGES

174

7b. NO. OF REFS

9

CONTRACT OR GRANT NO.

9a. ORIGINATOR'S REPORT NUMBER(S)

PROJECT NO.

9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)

DISTRIBUTION STATEMENT

Approved for public release; distribution unlimited.

SUPPLEMENTARY NOTES

12. SPONSORING MILITARY ACTIVITY

Naval Postgraduate School
Monterey, California 93940

ABSTRACT

A finite element formulation for solving axisymmetric transient heat conduction and thermal stress problems is developed in this thesis. The governing equations of uncoupled, linear, isotropic thermoelasticity are discretized using quadratic isoparametric elements. A FORTRAN IV program, using double precision arithmetic, is presented. Compact storage techniques for banded symmetric matrices are used.

Comparisons between exact and computer solutions demonstrate close agreement for a number of test problems. Detailed instructions for using the program are included.

KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
AXISYMMETRIC TRANSIENT HEAT CONDUCTION						
FINITE ELEMENT HEAT CONDUCTION						
FINITE ELEMENT TRANSIENT THERMAL STRESSES						
AXISYMMETRIC TRANSIENT THERMAL STRESSES						

Thesis

134923

B1735 Bakhshandehpour,
c.1 Finite element solu-
tion for axisymmetric
transient thermal stres-
ses.

19 OCT 72
9 APR 73
17 OCT 74

20468
20901
23427

Thesis

134923

B1735 Bakhshandehpour
c.1 Finite element solu-
tion for axisymmetric
transient thermal stres-
ses.

thesB1735

Finite element solution for axisymmetric



3 2768 001 91212 4

DUDLEY KNOX LIBRARY